Integrating Acquisition with Software and Systems Engineering: Providing More Structured Guidance to Better Satisfy the Needs of Users

Lt. Col. Joe Jarzombek
ESIP Director

With continued emphasis on acquisition reform initiatives, the Department of Defense and services acknowledge the significant role of acquisition organizations in delivering capabilities. They now have an opportunity to provide more structure to guide and assess program offices in maturing their acquisition capabilities. The need is evidenced by inspection and audit agency reports that have attempted to address why so many software-intensive systems have failed operational tests. Indeed, some of the findings and recommendations deal with needed changes to processes and practices within the acquisition organizations. Recognizing the need, some government program offices, both in the acquiring and sustaining phases of the life cycle, are now using the Software Acquisition (SA) Capability Maturity Model (CMM)SM as a framework to guide and assess their internal activities.

A Level 3 development effort coupled with a Level 1 acquiring effort often equates to a Level 1 delivery capability to the end user; yet the Level 3 developer is often blamed, and the Software (SW) CMM is cited as inadequate. The reality is that an “immature” acquirer can force poor practices upon the developing organization, and domain expertise is important, both in the acquiring and developing organizations. Integrated product teams (IPT) offer one of the better forums for bringing the developer and acquirer together, and there is an opportunity to offer more help to guide and assess the effectiveness of such forums. By packaging best practices, CMMs serve as guides for process improvement.

IPT best practices are identified in the draft integrated product and process development (IPPD) CMM that is a source model for the Office of the Secretary of Defense-sponsored CMM Integration (CMMI) product suite. Software CMM version 2.0, draft C and Systems Engineering (SE) EIA 731 are the other two source models in the CMMI.

However, even with these three models, there is a gap in coverage of some acquisition processes and practices that are critical to the delivery of products to the end user. At Software Technology Conference ’99, Dr. Dolores Etter, Deputy Under Secretary of Defense (Science and Technology) noted the need to determine how we include acquisition within CMMI.

Many of the system acquisition best practices are captured in the SA-CMM, and that model, coupled with practices identified in acquisition reform initiatives, offers an effective starting point for merging system acquisition within the CMMI. SW and SE models include many key acquisition process areas such as resource management, requirements management, planning, subcontractor management, monitor and control, and configuration management. As a guide to better enable program teams in meeting user needs—including certification of the systems for operational safety, suitability, and effectiveness—a complete CMMI needs to include other acquisition and IPPD processes to provide the remaining relevant functions that are vital to delivering capabilities. Additional processes are needed to cover practices associated with supplier capability evaluations, transition for product deployment and support, product life cycle and product lines definition and management, external quality management, contract management, work environment management, and rigorous reviews of supplier project plans and test plans and user requirements documents. Many of these processes are addressed in this special software acquisition issue of CROSSTALK.

Some guidance is needed to enable acquiring organizations to know how effective they are in performing their functions. While it may be more than a year before the CMMI includes acquisition, interested organizations can now take advantage of the Federal Aviation Administration’s integrated CMM (iCMM) to guide enterprise-wide process improvement since it integrates SW-CMM, SE-CMM, and SA-CMM in a single model. As an alternative, acquiring organizations might simply use the SA-CMM as a framework and use acquisition reform best practices as extensions to the processes identified in the SA-CMM. As a minimum, acquisition organizations should use some framework to guide and assess their capabilities that are vital to delivering systems and products to the users*. Successful development efforts are very dependent upon acquisition capabilities and practices.◆

* Assistance in using models to guide and assess organizational capabilities is available through the Software Technology Support Center.

---

On the cover: Salt Lake graphic artist Brandon Scott used computer graphics to illustrate this month’s special issue on software acquisition.
On March 23, 1999, Dr. Jacques Gansler, the Under Secretary of Defense (Acquisition and Technology), added acquisition software oversight, along with management oversight for the Software Engineering Institute (SEI), to my other responsibilities as Deputy Under Secretary of Defense (Science and Technology). Given that half of software projects double their original cost estimates, projects slip an average of 36 months, and one third of software projects are cancelled, I think this new role is one not only challenging, but especially critical to providing new and upgraded warfighting capabilities to our warfighters.

Software is pervasive. It truly is the new physical infrastructure. We are more dependent on software than ever, and software is becoming more complex. In our weapon and support systems, it is the source of enhanced functionality, the key to system modernization, the technology we use to model new system concepts, and the basis of our advanced simulation-based training and doctrine development. It is our competitive edge that enables our systems to interoperate in new and novel ways.

In my keynote talk at the Software Technology Conference (STC) on May 3, I outlined four areas in which I intend to focus. My first focus area is discipline. We know the components of good software development practice; requiring discipline in every step of this process is necessary. One way to do this is to use the SEI’s Capability Maturity Model for Software. We also know what kinds of metrics to collect and how to use them to manage software development and acquisition efforts. One example is the practical software measurement initiative. And we have many Department of Defense (DoD) organizations that are dedicated to providing the DoD acquisition community with software engineering expertise. One example is the parent organization of CrossTalk, the Software Technology Support Center. Collectively we need to insist on disciplined approaches to acquire our software-intensive systems.

My second focus area is collaboration. I have asked SEI to organize a workshop for the DoD organizations that are involved in improving the DoD’s collective ability to acquire software. I was initially aware of five to eight groups, and was surprised and delighted to find that approximately 30 organizations have stepped forward. This group of organizations provides a baseline capability to share information, facilitate use of common products and services, and recommend policy improvements that will help us all. We need to view ourselves as a community and work together as a community to improve.

My third area of focus is workforce development. We are fortunate to have a dedicated and talented workforce. This was made evident to me by the large number of attendees at STC this year and in my side meetings. I want to help ensure that our workforce is able to receive the professional and continuing education it needs to stay current. I also am very concerned that the DoD has the ability to act competitively in hiring and maintaining a world-class group of professionals. People are our strength, and that is an asset that needs our attention and commitment.

My final focus area is science and technology. In addition to my software role, I also manage the DoD’s science and technology programs. The rapid rate of advancement in technology and the unique issues faced when introducing those new technologies into our organizations and systems is a familiar challenge. I hope to make those technologies available to the software community as rapidly as possible. I also believe that a focused investment in science and technology based on the needs of the DoD software engineering community is mandatory to help us continue to improve.

I am committed to working with you to improve our collective ability to acquire software-intensive systems on time, within cost estimates, and with the desired functionality. I am pleased to see that this issue of CrossTalk is focusing on acquisition.
Using Contractor Capability Evaluations to Reduce Software Development Risk

Bonnie R. Troup
The Aerospace Corp.
Brian P. Gallagher
Software Engineering Institute

One aspect of software development risk is the development contractor's ability to deliver software within specified goals. One approach to mitigating this risk is to select the development contractor with the strongest software development capabilities. This paper will present and compare two methods used at Space and Missile Systems Center (SMC) and the National Reconnaissance Office (NRO) during source selection to evaluate a contractor's software development capability with the goal of selecting a mature software development contractor.

Despite the best efforts of software acquisition professionals, and the increased commitment of maturing acquisition organizations to deliver promised software capabilities, large software-intensive systems frequently do not meet the cost, schedule, and performance objectives of the acquiring organization and end user. As many as 75% of all large-scale, custom software-intensive systems fail [1]. The primary reason is immature management practices [2].

Given this poor record, and an increasing demand on software capabilities, how do software acquirers reduce this risk? One approach is to choose a contractor mature enough to manage the software project; a technique to aid in this choice is using a contractor capability evaluation during source selection. Such evaluations provide acquirers with information to discriminate among contractors by assessing their ability to deliver such systems within cost, schedule, and performance objectives. The evaluation results can reduce the risks by helping the acquirer identify weaknesses in a developer's software development process before hiring them.

Purpose of Contractor Capability Evaluations

Contractor capability evaluations are formal, systematic methods that employ defined models for assessing a contractor's software development process. These methods are used to identify strengths, weaknesses, and risks related to a contractor's defined or proposed software process. They also can be used to compare a contractor's defined or proposed software process with its actual software process in use on a given program.

Formal evaluations are performed by an external organization and the acquisition organization receives feedback on the evaluation. When an evaluation is done during source selection, evaluation results are a key discriminator to decision makers and increase the likelihood of selecting a contractor capable of developing the required software within program constraints. An acquirer also can use formal evaluations to identify risks inherent in the contractor's approach and to facilitate managing these risks beginning at contract award.

Success of contractor capability evaluations during a source selection assumes an evaluation of the contractor's process, commitment to using the process on the proposed project as evidenced in contractually binding documents like the Integrated Master Plan (IMP), and incentives for following and improving the proposed process during contract execution. Evaluation and commitment are determined during the source selection, and the incentive is generated by encouragement from the acquirer's leadership during contract execution and the appropriate use of award fee.

Capability Evaluations and Acquisition Reform

With the advent of the acquisition reform "lightning bolts" [3], the Department of Defense (DoD) significantly increased its emphasis on risk management, early communication with project bidders, and use of past performance in source selection, while substantially reducing the scope of contractual requirements and oversight on new programs. This reduction in oversight heightens the importance of selecting a contractor capable of reaching cost, schedule, and performance objectives and following a well-defined process while meeting these objectives. As evidenced in DoD 5000.2-R, paragraph 4.3.5.5 [4], selection of a contractor with "a demonstrable, mature software development capability and process" is mandatory on all major defense acquisition programs.

Capability evaluations are consistent with the objectives of acquisition reform. Evaluation results provide early software development risk identification, thereby supporting the role of risk management in acquisition decisions. The evaluation process establishes a beneficial communication with the contractors' software organizations, beginning with the early phases of the acquisition, and supports the use of contractor-defined or commercial processes in lieu of contractually dictated standards and processes. Finally, capability evaluations request evidence of past use of proposed processes, supporting assessment of past performance.

Primary Capability Evaluation Techniques

The Software Development Capability Evaluation (SDCE) and the Software
Capability Evaluation℠ (SCE) are two alternative, formal methods that the SM C and NRO predominantly use to evaluate the software development capability of contractors. The SD CE is the primary evaluation method in use at SM C; the SCE is used by the NRO and other government organizations.

The Software Development Capability Evaluation

Developed solely for use during source selections, the SD CE was created in 1993 by an Air Force Materiel Command Acquisition and Process Action Team (AFMC-PAT), which included participants from government, industry, and Federally Funded Research and Development Centers (FFRDCs). The SD CE is based on the Software Development Capability/Capacity Review (SDC/CR), developed by Aeronautical Systems Center, and on early versions of the SCE. The method is documented in AFMC pamphlet 63-103, “Software Development Capability Evaluation” [5] and discussed in “Software Development Capability Evaluation: An Integrated Systems and Software Approach” [6]. Guidelines for application of the SD CE, based on lessons learned and technology updates, are documented in Aerospace Technical Report TR-98(8550)-1 [7].

Acquisition philosophy recommends using capability evaluations to reduce software development risk; the SD CE supports this philosophy by enabling acquirers to consistently evaluate software development contractors for proven plans, processes, methods, and tools. First, the proposed development approach is assessed by evaluating a contractor’s written SD CE responses against the documented SD CE criteria. These responses are cross-checked with other portions of the proposal to determine consistency. The contractor’s commitment to follow the proposed approach is also assessed by comparing material in the SD CE responses with the wording contained in any contractually binding documents. Examples of these documents include the Software Development Plan, IMP, and Work Breakdown Structure. Lastly, using experiential evidence supplied by the contractor, the evaluation weights a contractor’s past experience against its proposed approach; if the contractor proffers a new approach, an analysis of the rationale supplied for that approach is performed.

Figure 1 illustrates the SD CE process. Planning for the evaluation is critical and includes tailoring the SD CE model and process for an individual acquisition. The tailored set of questions and criteria, and instructions for completing the SD CE, are developed and incorporated into the request for proposal (RFP). The contractors provide their responses and evidence of past performance for the acquisition team to review. If deficiencies are documented in evaluation notices (ENs) and provided to the contractor, strengths, weaknesses, and risks are established and integrated into the source selection. The SD CE results for the selected contractor can be used as a basis for starting a risk management effort after the contract is awarded.

The Software Capability Evaluation

The Software Engineering Institute (SEI) developed the SCE to support source selections in major government acquisitions of software-intensive systems. The method was originally documented in A Method for Assessing the Software Engineering Capability of Contractors [8]. It was publicly baselined in SCE Version 1.5 Method Description [9] and the current version is documented in SCE Version 3.0 Method Description [10].

The SCE’s purpose is to provide results that support senior management decision making. These results can be used as a discriminator to select contractors during a formal source selection, and to help assess process growth during contract monitoring. The SCE process supports a disciplined process improvement effort.
program within the development organization. Results are obtained relative to the capability M aturity M odel for Software (SW-CMM) [11]. The SW-CMM is a public reference model used by software development organizations worldwide to improve their software processes. The SCE process uses the SW-CMM to identify the strengths, weaknesses, and risks of an organization’s existing software development process.

Figure 2 depicts the SCE process. As with the SDCE process, planning is critical to the success of the evaluation. The pre-evaluation activities include tailoring the SCE to meet the needs of the acquisition and defining the scope of the evaluation and on-site activities. On-site visits are mandatory for the SCE process and the information acquired during these visits is analyzed to determine the findings that are incorporated into the source selection.

Comparing SDCE and SCE Techniques

The SDCE and SCE methods have been used to evaluate a contractor’s ability to develop software-intensive systems. They gather information using a defined model and use evidence from existing projects to establish capability. Results are developed in terms of strengths, weaknesses, and risks, and both have a defined process for integrating these results into a source selection.

The differences in origin, focus, and use are shown in Figure 3. The key difference in this table is that the SDCE is focused on assessing the proposed process for a specific software-intensive project — the one under bid. The SCE focuses on assessing the processes used by the organization bidding on the contract, on similar projects under way or recently completed.

Differences in preparation and implementation are shown in Figure 4. The chief distinctions are the use of site visits and the basis for establishing findings. The SDCE primarily evaluates the contractor’s written answers to a tailored questionnaire and the documentation supporting the answers. Site visits are optional and performed, as necessary, to clarify contractor responses. To date, site visits have been done on less than half of the SDCEs for SMC. The SCE requires doing the evaluation at the contractor’s site. At the site visit, documentation from the projects selected for evaluation, organizational process documentation, and interviews of project personnel are used to establish findings.

Another marked difference between the SDCE and SCE, as shown in Figure 4, is in the basis for assessing a contractor’s software development process. Although both the SDCE and SCE evaluate the existence and use of processes, the SDCE requires evaluators to determine the quality of the processes as well. The SCE uses the premise that a well-defined and measured process is self-correcting and that as long as there is sufficient insight into the results of the process, the contractor is able to determine quality and correct for deficiencies.

Differences in Model Coverage

The SDCE model includes questions on the contractor’s use of a system/software engineering environment (S/SEE) and in technology areas such as artificial intelligence, distributed processing, and object-oriented techniques. When using these areas of the model, the evaluation team must include members with expertise in the selected technologies to determine the quality of the proposed process. The SCE uses the SW-CMM as its model, which does not include specific technology areas.

The SDCE model contains an area focused on systems engineering process; the SCE’s model includes inter-group coordination with systems engineering and software engineering, but does not have process areas specifically for the systems engineering process.

As shown in Figure 5, both methods encourage tailoring their respective models and evaluation processes for a specific acquisition. The SCE method includes

<table>
<thead>
<tr>
<th>Origin</th>
<th>SDCE</th>
<th>SCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Focus</td>
<td>Specific software-intensive project</td>
<td>Organizational software process capabilities</td>
</tr>
<tr>
<td>Intended use</td>
<td>Source selection</td>
<td>Source selection and contract monitoring</td>
</tr>
<tr>
<td>Primary users</td>
<td>SMC, NRO, and ASC</td>
<td>Government, commercial, international</td>
</tr>
</tbody>
</table>

Figure 3. Differences: Origin, focus, and use.

<table>
<thead>
<tr>
<th>Questionnaire size</th>
<th>SDCE</th>
<th>SCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Questionnaire size</td>
<td>700+ questions; usually tailored to &lt;100</td>
<td>100 questions; some may be “NA”</td>
</tr>
<tr>
<td>Questionnaire responses</td>
<td>Essay with supporting data</td>
<td>Yes / No; comment required for “Yes”</td>
</tr>
<tr>
<td>Site visit</td>
<td>Optional; no defined process</td>
<td>Mandatory; well-defined process</td>
</tr>
<tr>
<td>Results established by</td>
<td>Questionnaire responses and optional site visits</td>
<td>Site visits; not from questionnaire responses</td>
</tr>
<tr>
<td>Assessment basis</td>
<td>Process existence, use, and quality</td>
<td>Process existence and use only</td>
</tr>
<tr>
<td>Guidelines for tailoring</td>
<td>SDCE</td>
<td>SCE</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------</td>
<td>-----</td>
</tr>
<tr>
<td>Tailoring risk</td>
<td>Not acknowledged</td>
<td>“Appraisal Risk” identified, documented and accepted</td>
</tr>
<tr>
<td>Tailoring decisions</td>
<td>No experience required</td>
<td>Restricted to trained evaluators</td>
</tr>
<tr>
<td>Model tailoring</td>
<td>Extensive tailoring required</td>
<td>Depth-oriented and/or breadth-oriented</td>
</tr>
<tr>
<td>Process tailoring</td>
<td>Permitted</td>
<td>Constrain number of projects and site visits</td>
</tr>
</tbody>
</table>

Figure 5. Differences: Tailoring approach.

extensive guidelines for tailoring out (removing) model components and modifying the evaluation process, which requires documenting risks the evaluation team assumes with each tailoring decision. Although the SDCE does not have specific tailoring guidelines, the planning process necessitates tailoring the questionnaire to focus on the project risks.

While the SDCE and SCE methods differ in several key areas, they are both powerful tools that have been used successfully to discriminate between contractors based upon their software development capabilities.

Summary
In the source selection environment, contractor capability evaluations can assist in identifying the contractor with the best software capability and experience. They facilitate software risk identification early in a program's life cycle and provide an in-depth look at potential high-risk areas. The acquirer's use of these methods highlights to the developers the importance of using mature software development processes on the projects they bid and encourages them to develop good processes early in the program.

Contractor capability evaluation is an important activity at SMC and NRO. The SDCE is the primary evaluation method used at SMC, while the SCE is popular with industry and is the primary method used by other government organizations.

While contractor capability evaluations are recognized as an acquisition best practice and are of immense value when acquiring a software-intensive system, a high-quality evaluation is resource- and time-intensive for the government and the contractor, and should be used with discretion. ◆

About the Authors
Bonnie R. Troup is an engineering specialist with The Aerospace Corp. She has more than 12 years of experience in software engineering and software acquisition management; she is the corporate focal point for the SDCE. During her employment at TRW, Troup was a project engineer and software engineering lead, where she participated in an SCE. Her work at Aerospace includes acquisition support for EELV, DMSP, and Mlistar, and coordinating SDCE efforts for several program offices. She received her bachelor of science degree in applied mathematics from U.C. Riverside, and a master of science degree in computer science from Loyola Marymount University.

Brian P. Gallagher is a member of the technical staff with the SEI. He has more than 14 years of experience in software engineering and software acquisition management; he is working within the Software Engineering and Process Management group at the SEI.

Gallagher was previously employed with The Aerospace Corp., where he worked as a software acquisition and engineering advisor for several SM C and N RO projects. He is a retired Air Force officer. He received his bachelor of science degree in management information systems from Florida Institute of Technology and a master of science degree in computer science/software engineering from Florida Institute of Technology.

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pa. 15213-3890
Voice: 412-268-7157
Fax: 412-268-5758
E-Mail: bg@sei.cmu.edu

References


Resources
SDCE Resources
Los Angeles Air Force Base Web site
http://ax.laafb.af.mil/~shulmaner/sdec.html

SCE Resources
Software Engineering Institute Web site
http://www.sei.cmu.edu/
Contains SEI documents on SCE and CMM

CERT (Computer Emergency Response Team)
Conference ‘99
Dates: Aug. 30-Sept. 2, 1999
Location: Omaha, Neb.
Topic: Information Protection Expo and Conference.
Provides a forum for discussion on issues concerning computer security and asset protection in our open environment. Topics include computer viruses, intrusions on privacy, data corruption, data theft, Web site vandalism, and hackers.
Hosts: Omaha SPIN (Software Process Improvement Network); Greater Omaha Chapter of Association for Communications, Electronics, Intelligence and Information Systems Professionals; the Software Engineering Institute; Applied Management Institute; and the Sarpy County Chamber of Commerce.
Contact: Debbie Jacobs/Susan Stewart
Fax: 402-293-2907
E-mail: cert@omaha.com or diszone@aol.com
Internet: http://www.omaha.com/cert

ESEC/FSE ‘99 Joint 7th European Software Engineering Conference (ESEC) and 7th ACM SIGSOFT International Symposium on the Foundations of Software Engineering (FSE-7)
Dates: Sept. 6-10, 1999
Location: Toulouse, France
Topic: ESEC/FSE ‘99 will bring together researchers and practitioners of modern software engineering techniques to exchange new research results and reports related to traditional and emerging fields of software engineering. ESEC/FSE ‘99 also will include a program of tutorials and workshops on current topics in software engineering.
Sponsors: ACM, SIGSOFT, CEPI, ONERA, and ONERA
Internet: http://www.iam.unibe.ch/~esec99/

2nd USENIX Conference on Domain Specific Languages (DSL ’99)
Dates: Oct. 3-6, 1999
Location: Austin, Texas
Sponsors: USENIX, the Advanced Computing Systems Association, in cooperation with ACM SIGPLAN and SIGSOFT
Internet: http://www.usenix.org/events/dsl99/

The Second International Conference on The Unified Modeling Language — UML ‘99
Dates: Oct. 28-30, 1999
Location: Fort Collins, Colo.
Objective: UML ‘99 will bring together researchers in academia and industry who are developing processes, methods, techniques, and semantic foundations for the UML. The conference will provide a forum for discussing and evaluating promising approaches that will enhance the application of UML.
Sponsors: IEEE Computer Society Technical Committee on Complexity in Computing in cooperation with ACM SIGSOFT.
Contact: Robert France, conference chairman
Voice: 970-491-6356
Fax: 970-491-2466
E-mail: France@cs.colostate.edu

13th annual Conference on Software Engineering Education and Training (CSEE&T) 2000
Dates: March 6-8, 2000
Location: Austin, Texas
Theme: Software Engineering Coming of Age
Submission Date: Sept. 17, 1999
Contact: Susan A. Engel
Voice: 806-742-3527
Fax: 806-742-3519
E-mail: mengel@ttu.edu
Internet: http://www.se.cs.ttu.edu/mengel

Contains AFM CSCE Pamphlet Vol. 1 and 2
Effective Acquirer/ Supplier Software Document Reviews

Gregory T. Daich
Software Technology Support Center

With all the quality initiatives of the last several decades, the software industry still continues to produce too many poor quality systems that are over budget and very late. Part of this problem can be attributed to software acquirers not understanding what they want and what they get at each stage of development. Improving the effectiveness of joint acquirer/supplier software document reviews can result in significantly less rework in later development phases. Project plans, requirements, and test plans are particularly important documents that acquirers should review using a more rigorous process than is practiced by many acquisition organizations.

What is in a Name?
Not all reviews are alike. What some people call a review (or a technical review), others call a free-for-all, come as you are, anyway you like it, gab session. Another name for some of these types of meetings is an informal design brainstorming session.

What some call a software inspection (or a peer review), others call a practice session for the latest speed-reading contest. Others call a formal review (or structured walk-through), others call a waste of time. Often valuable process and product information gets ignored or thrown away before it can be used to improve practices and show value for the effort expended.

Finally, what some call an audit (or software quality assurance review), others call a half-baked effort. Many reviews are severely handicapped due to missing or unavailable review resources or inadequate time.

Review processes can be called by a lot of different names. If your goal is to determine that a software work product is ready for the next phase of development or delivery, then you need review practices that:

• identify problems first before entertaining unsolicited corrections and determine ahead of time who will provide and review corrections and when those efforts will occur
• optimize the time spent by each participant through planning and focusing on what is important, thus helping them to be as effective and efficient as possible
• determine the benefits of each review and make use of critical information that can improve upstream processes and the review process itself
• bring all the key resources to bear on the effort such as subject matter experts; all references (source documents); and standards, rules, or checklists (telling what the organization has agreed a document should contain)
• provide the defined processes for planning and conducting reviews

Call your review process by whatever name you choose but insist that you have the above basic review practices in place in your organization. These concepts have significantly improved productivity and quality in numerous development and acquisition organizations [1].

Many managers incorrectly assume that their staff knows how to effectively and efficiently review technical documentation. With proper training many people have dramatically increased their skills to find and report defects. These people have learned where to look, how much time to spend, and what information to collect to optimize their review efforts.

What is the Problem Here?
The ultimate goal of most software product reviews is to determine product quality in order to verify that the product is ready for the next stage of development. However, acquisition organizations often do not have all the information they need to determine that readiness.

Reviews conducted by acquisition organizations generally fall into three main categories: project status management reviews, software product technical reviews, and product and process audits. These reviews depend on developer (supplier) participation to provide most of the inputs for these reviews; however, supplier participation during these reviews often leaves something to be desired. Suppliers do not want to make themselves look bad by finding problems that should have already been corrected. Thus their participation is self-relegated to simply answering questions from acquirers as needed. In other words, suppliers often do not fully participate as reviewers in joint acquirer/supplier reviews.

Developer reviews fall into the same basic categories as acquirer reviews (i.e. management reviews, technical reviews, and audits). Developer reviews should generally be more frequent and should look deeper into the software products than acquirer reviews. Unfortunately, many developer reviews are conducted haphazardly, if at all, with no data collected regarding critical defects. Thus, the potential to learn from past problems is lessened considerably. Also, a return on investment for the effort expended cannot usually be determined so no one can state objectively how well reviews are supporting development.

Acquirer reviews tend to involve higher volumes of material than developer reviews because it costs more to get suppliers and acquirers together. The result is that acquirers, with very little review support from developers and almost no useful information about prior conducted reviews, skim over the document. Acquirers simply do not have enough time to look deeply into the software products. Acquirer’s fears often come to pass in later stages of development when too many defects are found and schedules slip due to under-planned debugging, fixing, and
retesting. Would it not have been nice if only most of those defects could have been detected earlier or maybe even prevented?

What do the Standards Say?
Four key standards that address document reviews are worth considering by the Department of Defense (DoD) and commercial acquisition organizations. They are MIL-STD-1521B, EIA/IEEE J-Std-016-1995, IEEE/EIA 12207.0-1996, and IEEE Std 1028-1997 [2, 3, 4, 5]. The following subsections discuss these standards in order of their creation.

MIL-STD-1521B
MIL-STD-1521B is a military standard that has been cancelled for DoD use mostly because the government wanted to get out of the software standards business and use commercial standards [4]. This standard has been approved for public release but no further updates are expected. It lists a number of technical reviews and audits, some of which have become known as overkill for some projects. These include:

- system requirements review (SRR)
- system design review (SDR)
- software specification review (SSR)
- preliminary design review (PDR)
- critical design review (CDR)
- test readiness review (TRR)
- functional configuration audit (FCA)
- physical configuration audit (PCA)
- formal qualification review (FQR)
- production readiness review (PRR)

Software acquirers conduct these formal reviews and audits. However, MIL-STD-1521B does not say how to conduct them. These reviews often result in numerous comments and corrections to documents, many of which contain completely opposite views on the same text.

On the developer side, many contractors have spent a considerable amount of time preparing briefing slides for a MIL-STD-1521B technical review. This time could have been spent more productively doing internal peer-type of document reviews prior to the technical review. It has been apparent that some less progressive organizations have not conducted internal peer reviews with some projects I have been involved with. It appears that some acquirers have the privilege of being first to view some deliverables.

The Software Engineering Institute's (SEI) Software Capability Maturity Model (SW-CMM SM) advocates conducting peer reviews [6]. The SW-CMM M has a Key Process Area (KPA) that provides some guidance in conducting peer reviews. Software acquisition organizations also could benefit by adopting similar practices when participating in reviews.

The SEI has published another interesting Capability Maturity Model focused on software acquisition organizations (SA-CMM SM). However, there is not a specific KPA dedicated to reviews such as there is in the SW-CMM M [7]. Note that every one of the SA-CMM M KPA's depend on reviews of various types.

I am convinced some contractors could have avoided some embarrassment and subsequent rework had they focused more on finding defects using an effective review process. The MIL-STD-1521B technical review would have been more successful and the quality of the software products would have been better at delivery.

EIA/IEEE J-Std-016-1995
EIA/IEEE J-Std-016-1995 is an important, relatively new standard that many government and commercial organizations have adopted which was derived from MIL-STD-498 [8]. MIL-STD-498 is familiar with MIL-STD-498 since it was derived from other standards they have used to develop several government systems.

J-Std-016 states, "The activities and tasks in the standard tell what to do, not how to do it" [2]. According to the dictionary, "how" information is the manner or way in which we may do something whereas "what" is the "something" we are talking about [5]. Some standards focus on "what to do" in the interest of not constraining the user. Also, the sheer volume of some of these standards makes it impossible to delve into "how to do it." My position is that if an activity produces superior results and has general applicability, it should be standardized whether it is "what" or "how" guidance. We will discuss later the IEEE Std 1028, which provides some "how to do it" review process information. Additional review activities that have evolved through experience are recommended but are not covered in any commercial standard at this point.

J-Std-016 provides "uniform requirements for acquiring, developing, modifying, and documenting software." The only part of the standard that cannot be tailored and potentially removed for a given project are the tailoring requirements. Product evaluations are required for all software products built to satisfy a contract (i.e., an agreement between an acquirer and a developer). The standard was careful not to mention "review" in the discussion about software product evaluations, since analysis and testing are two other ways to evaluate a product.

The standard provides a list of criteria against which each type of software product can be evaluated. This is one of the best kept secrets of this standard, it seems, since some organizations are not using these criteria. Reviews tend to be the best method to verify that many of the listed criteria have been met. These criteria serve as a very useful starting point when defining review checklists.

While reviews could potentially be tailored out of the standard (and the organization's process) for a given project because they were deemed inappropriate, that would be like shooting yourself in the foot. Why would you want to inflict such a handicap on a project? Surely at least one documented, reviewable product will be produced in every development effort that will need to be evaluated (e.g., development plan or final software product). Since every project will produce some product that will require evaluation, we infer that product evaluations (reviews of work products) cannot rationally be tailored out.

J-Std-016 briefly discusses joint management reviews, which are similar to the MIL-STD-1521B technical reviews. J-Std-016 software product descriptions (templates) also provide some useful evaluation criteria. You should consider the applicability
of each section of a template for each project. This will help avoid the problem that I have termed the “factory approach to writing documents.” Often, people will take these templates and fill in the blanks with something just to complete it. The objectives of the project are not carefully considered as the document is written. Another problem is people taking a document from a prior project and replacing old parameters with new data. The real objectives of the project can easily be incorrectly biased toward the previous project.

**IEEE/EIA 12207.0-1996**

IEEE/EIA 12207.0-1996 is a new standard that brings it all together. Not only is the development process covered but so are other primary life cycle processes, including acquisition, supply, operation, and maintenance. Also, supplemental guides provide additional implementation information and example life cycle data (document content and references to document templates in dozens of other useful standards and guides) [9, 10].

Again, this standard does not provide “how to” information but references IEEE Std 1028-1997 for that information. However, there are useful lists of criteria provided for many types of documents.

**IEEE/EIA 12207.0-1996 addresses** supporting life cycle processes including:

- documentation
- configuration management
- quality assurance
- verification
- validation
- joint review
- audit
- problem resolution

A joint review between an acquirer and a supplier is required for many of the activities in the development process and other primary and supporting life cycle processes. Basically, all deliverable documents that could have quality problems should be considered for joint review. The verification supporting process mentions reviews as one of the verification activities, with analysis and test as other options. Verification tasks that could benefit from various types of reviews include:

- contract verification
- process verification
- requirements verification
- design verification
- code verification
- integration verification
- documentation verification

The next section discusses the IEEE Std 1028, IEEE Standard for software reviews, that provides some “how to” information that development and acquisition organizations should consider.

**IEEE Std-1028-1997**

This standard defines systematic review practices applicable to acquisition, supply, development, operation, and maintenance processes. It states, “This standard describes how to carry out a review. Other standards or local management define the context.

<table>
<thead>
<tr>
<th>Type of Review</th>
<th>Review Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Management Reviews</td>
<td>“The purpose of a management review is to monitor progress, determine the status of plans and schedules, confirm requirements and their system allocation, or evaluate the effectiveness of management approaches used to achieve fitness for purpose. Management reviews support decisions about corrective actions, changes in the allocation of resources, or changes to the scope of the project. Management reviews are carried out by, or on behalf of, the management personnel having direct responsibility for the system. Management reviews identify consistency with and deviations from plans, or inadequacies and inadequacies of management procedures. This examination may require more than one meeting. The examination need not address all aspects of the product.”</td>
</tr>
<tr>
<td>Technical Reviews</td>
<td>“The purpose of a technical review is to evaluate a software product by a team of qualified personnel to determine its suitability for its intended use and identify discrepancies from specifications and standards. It provides management with evidence to confirm whether: a) The software product conforms to its specifications b) The software product adheres to regulations, standards, guidelines, specifications, plans, and procedures applicable to the project c) Changes to the software product are properly implemented and affect only those system areas identified by the change specification”</td>
</tr>
<tr>
<td>Inspections</td>
<td>“The purpose of an inspection is to detect and identify software product anomalies. This is a systematic peer examination that: a) Verifies that the software product satisfies its specifications b) Verifies that the software product satisfies specified quality attributes c) Verifies that the software product conforms to applicable regulations, standards, guidelines, specifications, plans, and procedures d) Identifies deviations from standards and specifications e) Collects software engineering data (for example, anomaly and effort data) (optional) f) Uses the collected software engineering data to improve the inspection process itself and its supporting documentation (for example, checklists) (optional)”</td>
</tr>
<tr>
<td>Walk-Throughs</td>
<td>“The purpose of a systematic walk-through is to evaluate a software product. A walk-through may be held for the purpose of educating an audience regarding a software product. The major objectives are to: a) Find anomalies b) Improve the software product c) Consider alternative implementations d) Evaluate conformance to standards and specifications Other important objectives of the walk-through include exchange of techniques and style variations and training of the participants. A walk-through may point out several deficiencies (for example, efficiency and readability problems in the software product, modularity problems in design or code, or untestable specifications).”</td>
</tr>
<tr>
<td>Audits</td>
<td>“The purpose of a software audit is to provide an independent evaluation of conformance of software products and processes to applicable regulations, standards, guidelines, specifications, plans and procedures.”</td>
</tr>
</tbody>
</table>

Table 1. Purpose of reviews according to IEEE 1028-1997.
within which a review is performed, and the use made of the results of the review” [3]. It also says that it provides minimum acceptable requirements for systematic software reviews where systematic implies the following attributes:

a) team participation
b) documented results of the review
c) documented procedures for conducting the review

IEEE Std 1028 defines the basic processes for the following types of reviews:
- management reviews
- technical reviews
- inspections
- walk-throughs
- audits

Table 1 defines the purpose for each type of review. Many organizations will see common activities between their types of reviews and the reviews defined in this standard. However, there may be some significant activities in this standard that an organization has not adopted into its process that it should consider. If a review activity could result in finding more defects, if it could help an organization learn from past mistakes, or if it could better estimate the value of the efforts expended on reviews, then these activities may prove extremely beneficial.

Management reviews and audits have some similarities with the other types of reviews but also have unique characteristics. For example, a management review is the only type of review where financial status is examined and that is optional. Information from all types of reviews can be input to management reviews. However, document defects or anomalies are often found and recommendations have often been prepared by the time a management review is conducted.

Audits specifically review software processes in addition to software products. Some process information is gathered through interviews which are unique to audits. Also, the reports and the feedback are much different for audits than for the other reviews.

There are surprisingly only a few major distinguishing characteristics between technical reviews, inspections, and walk-throughs. Technical reviews basically are the same type of review as an inspection. However, inspections emphasize software product revisions as part of the review process, whereas technical reviews can conclude with a list of anomalies and recommendations. Technical reviews involve more people than just peers, such as management and acquirers. Inspections typically involve only peers, but this article addresses how inspections can involve acquirers and suppliers in joint reviews. The most significant difference between inspections and technical reviews is that inspections are more in-depth and usually cover a lower volume of materials.

Walk-throughs appear to be much less thorough than an inspection. More review materials are brought to bear with inspections than walk-throughs. However, more time is spent in walk-throughs discussing and investigating alternative solutions. This could be more of a problem than a help. If the walk-through diverges to mostly discussions about better design alternatives, then less time will be spent in finding and understanding problems before solutions are recommended. Every solution discussion should consider whether all reviewers are needed for that discussion.

Should an organization plan to conduct all of these types of reviews? Tom Gilb, in his book on software inspections, says that walk-throughs are for training [1]. He also says reviews are for gaining consensus, but the review Gilb is talking about is principally the IEEE Std 1028 management review does. Finally, he recommends that to find defects and determine document quality, inspections are the best method.

IEEE Std 1028 and this article treat reviews as a generic term that encompasses any type of document or process examination and evaluation. This standard stresses the importance of determining the objectives of each type of review by requiring this to be written as an entry criterion prior to conducting a review. There are several additional activities that Gilb and others advocate on top of this standard that are well-known in many practicing organizations. Some of the “how to do” information missing in IEEE Std 1028 include:

- monitoring inspection rates (e.g. pages reviewed per hour)
- providing useful reference citations (include page and section)
- sampling for inspections
- using entry criteria (e.g. number of defects found) to determine if a review meeting should be held
- using developer inspection data to give acquirers additional insight into software product quality
- using inspection practices as a mechanism for conducting joint technical reviews between acquirer and developer

These practices can significantly increase acquirer software document review effectiveness and efficiency. The next section outlines a process that a few acquisition organizations have used to help better determine software product quality and readiness for the next stage of development.

Note that the IEEE standards can be obtained by calling 1-800-678-IEEE.

What is an Acquirer/Supplier Software Document Review?

There are several types of acquisition-related software document reviews:

1. acquirer document review
2. contractor document review
3. joint document review
4. joint management review

Acquirer document reviews are reviews of documents that acquisition organizations write. These documents include policies, contracts, Statements of Work (SOWs), and various plans. In contrast, contractor document reviews are internal reviews of documents that may or may not be delivered to the acquirer. The software products that are delivered to the acquirer are of
particular interest because the acquirer usually must review and approve them. These documents include software development plans, requirements documents, design specifications, test plans, and user manuals.

The following two subsections discuss typical problems with acquirer document reviews and joint document reviews.

**Acquirer Document Review**

Acquirers and developers have achieved increased understanding of project deficiencies with the use of document inspections as described in the IEEE Std 1028-1997. Unfortunately, I have seen only a few acquisition organizations employ these types of inspections when reviewing software work products delivered by a contractor.

The techniques for reviewing acquirer-written documents often result in skimmed-over reviews, with individual reviewers obtaining little or no guidance on what to look for.

The underlying assumption is that you read, you find defects, you report them. What more is there to a review? If you ask for some clear objectives for a review, you can almost hear some people say, “You have been in this business how long? You should know how to review.” With little or no direction, here is an example of what can happen:

Ten people are asked to review a SOW. Two hundred issues get reported, with many issues completely opposite from others. Some reviewers choose to rewrite portions of the SOW, while others write several paragraphs of text explaining why they think something is wrong.

These review practices cost extra project time and do not often result in finding and fixing enough serious problems.

Acquisition organizations conducting inspections of documents they produce can have similar effects to what some developers have experienced. These effects include high quality documents, increased understanding of document content, and reduced amount of downstream rework for themselves and their contractors.

**Joint Document Review**

Sometimes acquirers believe they do not have the time to fully participate in a joint document review (i.e. technical review of deliverable software product) so they watch from behind the scenes. When the developer signs off that all evaluation criteria have been met, some acquirers accept it. This often results in acquirers feeling uncomfortable about the product not really being ready for the next stage of development. Acquirers do not have the objective quality information they need to assess readiness. Their fears are often confirmed when the system reaches later life cycle stages and too many defects start surfacing. For example, many products have reached operational test and evaluation (OT&E) only to fail miserably at that stage. You wonder how they made it out of the developer level of testing. Yet, the system to OT&E.

Most acquisition organizations in the DoD and many commercial organizations have experienced some downsizing, which often results in increased workloads and less time to manage and track the efforts of their contractors. DoD acquisition reform transfers more management and oversight responsibilities from acquisition organizations to contractors. But this does not lessen the need for acquirers to understand what they are buying and to approve what is delivered. More than ever, practical techniques and mechanisms are needed to gain improved insight into the quality of delivered software products in a shorter time than once permitted.

Sometimes, acquirers do not fully participate in document reviews because they lack specific knowledge. Can an acquirer always participate effectively in a technical review of a software product? Some acquirers knowingly or unknowingly focus on crossing “T’s” and dotting “I’s.” What constitutes a significant issue? The next two sections elaborate on these review problems.

**Subject Matter Experts**

Many people assume that subject matter experts (SM Es) will naturally be effective in reviewing a document related to their expertise. In other words, SM Es do not need to follow a process for reviewing a document, they will see all the defects and we will all be saved. However, humans have a difficult time keeping more than five to nine concepts in their short-term memory at one time. How can you expect a SM E to effectively review a document without looking at its references (sources, regulations, standards, guidelines, plans, and procedures)? Yet this is how many people review a document — looking only at the document and not at any of its references.

If a document under review is dependent on several references, those references should be checked to be sure that the document is correct and consistent with them. If a reviewer (SM E or not) does not take the time to check the references, he or she is missing an opportunity to find serious defects early which can be corrected before others encounter them.

What if a specific SM E is not available for a planned review? This is a main reason to use standards or rules for writing documents. We want to decide ahead of time what the doc-

![Figure 1. Joint document reviews — program office side](image-url)
units should contain. We can glean a lot of information from SMEs by encoding their knowledge, so to speak, in documents such as test plans. I have seen this problem in many types of documents. A document with no clearly defined objectives has a difficult time gaining buy-in from others because no one is sure what the document should do. Some process improvement-related documents (action plans or guides) often do not contain adequate citations of references, making their credibility questionable.

One of the most insidious of all problems is a document with no clear statement of objectives for the project and the document. Having seen this problem in many types of documents. A document with no clearly defined objectives has a difficult time gaining buy-in from its readers or users because no one is sure what the document should do. Some process improvement-related documents (action plans or guides) often do not contain adequate citations of references, making their credibility questionable.

An author is, by definition, an SME and we would not think of conducting a review without them. But other SMEs may not be available who could contribute to the review. Getting them involved early when establishing the objectives and general direction for a project can help alleviate some schedule problems when you might have a little more flexibility with the schedule. This will also help the project start effectively by getting early agreement from key personnel that you are headed in the right direction.

Finally, with a good set of rules (or standards) and checklists and an effective process, acquirers can be very productive in identifying document deficiencies in contractor-developed documents. This is true even if some acquirers do not yet know a great deal about the system. They must learn about it and they must sign off that a document is ready for the next stage of development.

The section below, What is the Recommendation?, outlines an effective acquirer/supplier software document review (joint document review) process. This process has helped acquirers find serious problems undetected by the developer and it has helped find more problems earlier than with traditional technical review practices.

**Major vs. Minor Defects**

A defect is considered serious or major if we estimate it could (not would) take more than an order-of-magnitude to fix later and correct its resulting consequences vs. fixing it now. Furthermore, fixing an actual software defect later often introduces more problems. Regression testing is done to assure that no new defects have been introduced in the unchanged parts of a system. This time should also be estimated when considering whether a document defect is major or minor. This perspective augments the definition of a major defect (anomaly) provided in IEEE Std 1028. This standard says that a major anomaly is one, “that would result in failure of the software product or an observable departure from specification.”

A minor defect will not likely require more time to fix later vs. now. IEEE Std 1028 says that a minor anomaly causes the software product to “deviate from relevant specifications but will not cause failure of the software product or an observable departure in performance.”

<table>
<thead>
<tr>
<th>Activity Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Review Draft</strong></td>
</tr>
<tr>
<td><strong>Approval Authority Reviews Issues</strong></td>
</tr>
<tr>
<td><strong>Verify that Defects were Resolved</strong></td>
</tr>
<tr>
<td><strong>Approval Authority Conducts Project Review</strong></td>
</tr>
</tbody>
</table>
What is the Recommendation?

This section describes an acquirer/supplier software document review (joint document review) process that can help acquisition organizations obtain useful software product quality information in order to make key decisions in a timely manner. This is an innovative systematic review process that implements powerful IEEE Std 1028 inspection technologies to provide critical document (and project) quality information. These practices can be used by acquisition organizations to knowledgeably approve documents and permit follow-on work to start. Figure 1 shows the major acquirer review activities and associated data. Figure 2 shows the major developer activities and associated data. The dashed lines represent internal activities and documents to either the program office or the contractor.

Table 2 discusses the acquirer document review activities. Table 3 discusses the developer document review activities. Table 4 discusses internal data to the program office, internal data to the contractor, and data transferred back and forth between the program office and the contractor.

Conclusion

As an industry, we need to look for ways to open up communication between acquirers and suppliers to willingly discuss problems and risks. The acquirer/supplier software document review (joint document review) process recommended in this article can do this by helping acquirers and developers gain better insight into document quality. Acquirers need more accurate quality information to determine readiness of developers to proceed to follow-on stages of development.

Both risk management and document inspections are recognized best practices by leading consultants in the industry. However, the traditional type of joint acquirer/supplier technical review does not qualify as a comprehensive, effective, and efficient IEEE Std 1028-type of document inspection. The traditional technical review is often a skim-over type of review that finds some problems but does not accurately assess a document’s quality. The IEEE Std 1028-type of inspection permits reviewers to delve deeper in representative document samples to obtain better insight into document quality. This information, coupled with inspection data from internal developer inspections, permits acquirers to make more informed document approval decisions.

The joint document review discussed in this article provides a mechanism shown to be effective with several contractor and acquisition organizations we have worked with. As with all process improvement efforts, there were growing pains during

<table>
<thead>
<tr>
<th>Activity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prepare Draft Document</td>
<td>The documents of particular interest that are prepared in this internal contractor activity are those that require approval by the program office for a particular development stage or for delivery. These include plans, requirements documents, designs specifications, source code, user manuals, etc.</td>
</tr>
<tr>
<td>Review Draft Document</td>
<td>This is an IEEE Std 1028 inspection type of review that shall include the author or author representative. The Draft_Document_Review_Report should be prepared and should be available for the program office to review. This activity should include a knowledgeable acquirer. Note that identified defects are corrected prior to delivery to the program office.</td>
</tr>
<tr>
<td>Update Deliverable</td>
<td>The contractor corrects the Draft_Deliverable to address all Defects_to_be_Corrected. This activity may need to occur repeatedly until the program office is satisfied and approves the Draft_Deliverable which then becomes the Approved_Deliverable.</td>
</tr>
</tbody>
</table>

Table 3. Joint document reviews — contractor activities

<table>
<thead>
<tr>
<th>Document</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Draft_Document</td>
<td>This is the contractor document that is ready to be reviewed (IEEE Std 1028 type of inspection) by the contractor. It is not yet ready for delivery.</td>
</tr>
<tr>
<td>Draft_Deliverable_Report</td>
<td>Since acquirers cannot attend all document reviews that contractors should perform, obtaining the statistics and defect information from the contractor document reviews could give useful insight into the contractor’s review process. The time spent and size of the document should be included in this report. This information can also help determine how much of the Draft_Deliverable to review.</td>
</tr>
<tr>
<td>Draft_Deliverable</td>
<td>This is the document that the contractor believes is ready for the next stage of development or for final delivery to the customer.</td>
</tr>
<tr>
<td>Program_Office_Issue_Log</td>
<td>This is an internal log of issues found by the acquisition organization and the contractor authors or author representatives. This log is not given to the contractor. It is ordered by most significant issues to least so program office management can quickly understand the most serious issues.</td>
</tr>
<tr>
<td>Deliverable_Review_Report</td>
<td>This is an internal report of the required review statistics. At a minimum, it includes the total amount of review time, number of major issues found, and number of pages reviewed. It is an internal document that’s not given to the contractor. An estimate of the savings in downstream rework should also be included.</td>
</tr>
<tr>
<td>Defects_to_be_Corrected</td>
<td>These are the most significant defects that must be corrected prior to approval of the deliverable. As a courtesy, it may include minor issues that can be cleaned-up as time permits.</td>
</tr>
<tr>
<td>Updated_Deliverable</td>
<td>This document has been corrected by the contractor and should have addressed all Defects_to_be_Corrected. If it didn’t, then the contractor will be asked to correct the document or the Defects_to_be_Corrected document will need to be changed.</td>
</tr>
<tr>
<td>Deliverable_Defect_Resolution_Summary</td>
<td>This document contains the status of each defect and summarizes the changes made to the Draft_Deliverable.</td>
</tr>
<tr>
<td>Project_Status</td>
<td>This information consists of normal project status information that may be required during the joint management review.</td>
</tr>
<tr>
<td>Approved_Deliverable</td>
<td>This, of course, is the approved deliverable that now meets the acquirer’s requirements as best as the acquirer and the contractor can tell at the time of approval. Some projects may need to move out and use a Draft_Deliverable prior to approval. This should be coordinated with the program office.</td>
</tr>
</tbody>
</table>

Table 4. Joint document reviews — program office documents
implementation. We initially had some questions about the level of contractor participation we would obtain. However, contractors, though initially reluctant, actively supported these joint document reviews. It was almost surprising how they willingly reported several significant issues. They wanted to actively participate for the benefit of all. I am convinced that these reviews even made a difference in how the contractor will conduct future internal document inspections.

"Mature" contractors want to find and fix problems to save downstream rework effort. "Mature" acquisition organizations want to approve documents that have been reviewed carefully with effective document review practices. This can ultimately result in acquirers being able to better handle their total workload.

If you would like more information about the joint document reviews (inspections) discussed in this article, please feel free to contact the author. A related article discussing demonstrating to your organization the power of these effective reviews was written for CROSSTALK in June 1999 [12].

**About the Author**

Gregory T. Daich is a senior software engineer with Science Applications International Corp. under contract with the Software Technology Support Center (STSC). He supports STSC’s Software Quality and Test Group with more than 22 years experience in developing and testing software. He has taught more than 60 software test, document inspection, and process improvement seminars in the last five years.

Daich consults with government and commercial organizations on improving the effectiveness and efficiency of software quality practices. His consulting approach coordinates formal document inspections with analysis of test work products to identify opportunities for software test process improvement. These practices have also been applied in supporting and testing Year 2000 upgrades. He is the principal author for several guidebooks and workshops for conducting Year 2000 compliance projects. These guidebooks address corporate- and project-level compliance efforts as well as Year 2000 desktop (PC and Macintosh) software compliance.

Daich has a master’s degree in computer science from the University of Utah and a bachelor’s degree in mathematics from Weber State University.

**References**

Product Line Acquisition in the DoD: the Promises, the Challenges

Lawrence G. Jones
Software Engineering Institute

Industrial use of software product line technology has resulted in some impressive savings while also improving product quality and delivery time. Although there has been some successful use of this technology within the Department of Defense (DoD), there are special challenges. This article reports the results of two DoD product line workshops in which important issues and successful practices were shared.

Do you find yourself continually acquiring software-intensive systems that are similar to ones you have paid for in the past? Do you wish you could use your scarce resources to buy what is truly new functionality without also having to pay for redevelopment of essentially the same old solutions? Some solutions to these frustrations are found in a maturing technology that is ripe for exploitation — software product line practice. Through this technology, a growing number of commercial organizations are reporting impressive reductions in costs, faster delivery of mission capability, and improved quality. To help transition this promising technology to the DoD, the Software Engineering Institute (SEI) established the Product Line Systems Program.

While this technology has great promise and relevance for DoD needs, key issues must be addressed to take advantage of this successful commercial practice. In this paper we will provide some insight into this important technology and its application within the DoD. After providing some background, including key concepts and relevance to the DoD, we will present some practical results from two SEI DoD product line workshops. By sharing the experience of successful DoD product line practice, we hope to allow others to take advantage of this promising technology.

Key Concepts

The field of product lines is new enough to offer different definitions for similar concepts. The SEI settled on a definition that brings together the key intent of these sometimes competing definitions. We define a product line to be a group of products sharing a common, managed set of features that satisfy specific needs of a selected market or mission. For example, a telecommunications company may offer a number of cellular phones that share a similar market strategy and an application domain, thus making up a product line.

The products in a software product line can best be leveraged when they share a common architecture that is used to structure components from which the products are built.

The architecture and components are central to the set of core assets, sometimes referred to as the platform, used to construct and evolve the products in the product line. In other words, a software product line can best be leveraged by managing it as a product family, which is a set of related systems built from a common set of assets. For example, if the product line of cellular phones is built from a common architecture and set of common components, it is managed as a product family. When we refer to a product line, we always mean a software product line built as a product family. This particular use of terminology is not nearly as important to us as the underlying concepts involved — using a common asset base in the production of a set of related products.

Product line practice is therefore the systematic use of software assets to modify, assemble, instantiate, or generate the multiple products that constitute a product line. Product line practice involves strategic, large-grained reuse as a business enabler.

Since software reuse is not a new concept, a key difference from earlier, less successful reuse efforts is that early efforts focused on small-grained reuse of code. The cost of creating and using these small-grained assets often outweighed the modest gains. Over the years, reuse technology has evolved to focus on progressively larger-grained assets. Today, the state of the art is to reuse strategic, large-grained assets unified by a software architecture. Using this approach, reuse can result in remarkable efficiency and productivity improvements and time economies [1, 2]. In combination with the known benefits of process improvement and technology innovation, systematic reuse through a product line approach offers great promise to software development and acquisition organizations.

Benefits of a Product Line Approach

A number of organizations have gained order-of-magnitude improvements in efficiency, productivity, and quality through a product line approach. Often more important than cost savings is the fact that product line practice enables an organization to get its product to field more rapidly. As Robert H.arrison, Naval Systems Warfare Center, stated, “The right answer delivered late is the wrong answer” [2].

A few examples of the reported benefits follow. The Swedish naval defense contractor, CelsiusTech, reported a reversal in the hardware-to-software cost ratio, 35:65 to 60:20, that now favors the software [1]. Hewlett Packard has collected substantial metrics showing two to seven times cycle time improvements with product lines. Motorola has shown a four times cycle time improvement with 80 percent reuse. Among other commercial domains that have shown equally dramatic results are air traffic control (Thompson), commercial bank systems (Alltel), engines (Cummins), telecommunication systems (Ericson, Nokia, Lucent, AT&T), and college registration systems (Buzzee).

The reported benefits are compelling, but what do you do when you engage in a product line approach?
The Essential Activities of a Product Line Approach

At its essence, fielding a product line involves core asset development or acquisition, and product development or acquisition using those core assets [3]. These two activities can occur in either order, or most commonly in concert with each other. Core asset development/acquisition has been traditionally referred to as domain engineering. Product development/acquisition from core assets often is called application engineering. The entire process is staffed, orchestrated, tracked, and coordinated by management. Figure 1 illustrates this triad of essential activities. The iteration symbol at the center represents the decision processes that coordinate the activities.

The bi-directional arrows indicate not only that core assets are used to develop products, but that revisions or even new core assets might, and most often do, evolve out of product development. The diagram does not specify which part of the diagram is entered first. In some contexts, already-existing products are mined for generic assets that are then migrated into a product line. At other times, the core assets may be developed or procured first in order to produce a set of products that is merely envisioned and does not yet exist.

There is a strong feedback loop between the core assets and products. Core assets are refreshed as new products are developed. In addition, the core assets’ value is realized through products that are developed from them. As a result, the core assets are made more generic by considering potential new products on the horizon. Finally, both the core asset and the product development or acquisition are iterative, as illustrated in Figure 1.

While it is evident that product line practice calls for a new technical approach, new nontechnical and business practices are equally crucial. There is a constant need for strong visionary management to invest resources in the development or acquisition of the core assets and to develop the cultural change to view new products in the context of the core assets. As we will see, the nontechnical challenges may be the most significant for the DoD.

Relevance and Challenges to the DoD

Some might ask what these largely commercial practices have to do with the DoD. First, there is no doubt that commonality of DoD requirements is abundant. For example, many DoD organizations have developed their own payroll systems, budgeting systems, and command and control systems that are essentially duplicates of others. In response to this commonality of requirements, there is a growing recognition within the DoD that new acquisition approaches leveraging best commercial practices must be implemented [2]. At the top DoD policy levels, acquisition reform from DoD Directive 5000.1 and DoD Regulation 5000.2-R have focused on using these best practices to reduce cost, schedule, and technical risks, advance architecture-based approaches to reuse and support open systems, interoperability, and commercial-off-the-shelf software (COTS). Former and current top-level policy makers have expressed the importance of the DoD using innovative, commercially proven practices to reduce cycle time, improve quality, reduce cost, improve efficiency, and reduce technical risks. These facts establish a clear linkage between DoD needs, policy, and product line practice [4].

While we know that product line practice works in industry, many attempts to emulate this success within the DoD have encountered problems. There are those who believe that there are inherent structural impediments against product line practice within DoD. While the technical challenges are not to be underestimated, even if they are solved, significant nontechnical barriers must be addressed [5]. In the DoD, many of these nontechnical issues translate into acquisition-related issues. Yet there is hope. There have been several reuse efforts within the DoD, and there are examples where the systematic reuse and horizontal leverage characteristic of a product line approach have occurred and are occurring [2].

Why have some attempts succeeded where several have failed? The successful organizations have found ways to identify and address some of the key acquisition-related issues. In the next section we present the results of two hands-on DoD workshops in which many issues and some answers were identified. Because this is a relatively new endeavor, many questions are unresolved. However, there have been enough successes to provide some optimism for the future.

Some Issues and Strategies for the DoD – Product Line Workshop Results

The SEI’s Defense Product Line Practice workshops were held in March 1998 [2] and March 1999 [6]. Their purpose was to identify industry-wide best practices in software product lines, to share DoD product line experience, to explore the technical and nontechnical issues involved, and to discuss ways in which the current gap between commercial best practice and DoD practice can be bridged. In each workshop, more than 30 participants represented...
joint agencies, all services, non-DoD agencies, and industry. All participants had experience with product lines or other strategic reuse approaches.

The participants formed working groups to consider the general areas of software engineering, technical management, and organization management for both acquisition organizations and contractors. After identifying the specific practices to discuss, the general approach of each working group was to:
- describe the practices in a DoD context
- identify barriers for implementing the practices within the DoD
- identify strategies to overcome those barriers

Following the same structure, we present highlights of the results most directly related to a DoD acquisition organization considering adopting a product line approach. Results from both workshops are summarized here. The practices covered are:
- building and communicating a business case
- developing and implementing a product line concept of operations
- achieving the right organizational structure
- providing an appropriate funding model
- developing and implementing an acquisition strategy
- contractor interface

Please note that this list is not purported to be an exhaustive list of all the issues. However, these are critical issues the participants were able to address in the context of the workshop.

Building and Communicating a Business Case

Given sound business goals as a basis for evaluation, a business case will play a strategic role in deciding whether a product line approach makes sense for a DoD organization. The current environment of downsizing and escalating demands for "better, faster, cheaper" system development may make building a business case more straightforward. While data from outside organizations may be useful to initiate concept exploration, hard evidence obtained from pilots within the organization is essential.

Participants identified the following prerequisites for building the business case:
- selectivity about where and when to apply a product line approach
- multiple mission areas may need different approaches
- solid justification, including anticipated savings or payback for candidate systems
- incentives for achieving efficiency

Some of the significant barriers to implementing this practice in the DoD relation to organizational structure and funding models. These will be discussed later in this section.

One mitigation strategy is to include a rough draft of the product line concept of operations with the business case to provide insight into how the product line concept will work within the organization. This will help to substantiate the considerations that are valid for the organization.

Developing and Implementing a Product Line Concept of Operations

Once a business case has been established to support a product line approach, it is important to begin creation of a product line concept of operations (CONOPS) to define how the implementation will be accomplished. The CONOPS will be best developed in an iterative fashion. As noted in the previous section, a draft CONOPS can be an important vehicle to identify key issues that must be resolved, such as which organizations will participate, how the approach will be funded, and processes and structures for initiating and sustaining the approach. As these issues are resolved, the CONOPS can be refined.

A fully developed CONOPS identifies product line stakeholders and clearly describes their roles and responsibilities. Typical contents include appropriate mechanisms for sustaining the product line over its life cycle, improving feedback mechanisms, customer interface, and other support functions essential for long-term success. The CONOPS should address the operation of both the acquisition and development groups, as well as the role of the product line architecture.

Workshop participants stressed that the key pitfall in creating a CONOPS was to adopt a "Big Bang" strategy that was too grandiose. Such a strategy ignores the reality that a product line approach should evolve incrementally, preferably from grassroots support that builds upon initial successes within the organization. Since the CONOPS describes how a product line approach will work in a particular environment, the document can serve as a practical way to identify a wide range of barriers and how the organization will mitigate them.

The SEI has developed guidance for the CONOPS creation based on experience with several government organizations. This may be found on the SEI Web site.

Achieving the Right Organizational Structure

A key part of a product line CONOPS is a description of the organizational structures involved. The workshop participants agreed that achieving the right organizational structure is one of the greatest challenges in implementing a product line approach. Implementing a product line approach is dependent on managing horizontally (i.e. in a matrix mode) across projects to produce products that are part of a family built around a common architecture and core set of assets, as well as managing vertically to create individual products. This presents a real challenge for DoD organizations that are traditionally highly "stovepiped" with regard to their sponsorship, project structure, funding, resources, contracting, and reward system. As one participant stated, "we [in the DoD] are horizontally challenged."

A primary consideration in a product line approach is structuring the organizational units responsible for developing/acquiring and sustaining the core assets vs. those responsible for developing/acquiring derivative products using the core assets. These organizational considerations raise many questions about control and funding of the architecture and other core assets, how the core assets will be responsive to project-specific requirements, and support for acquisition of assets and products.

The wrong organizational structure can defeat solid product line technology and processes. Moreover, achieving the right organizational structure involves
both determining the appropriate structure and an effective strategy to implement it. The definition of the right organizational structure may also change as the product line matures. The challenge in creating a suitable organizational structure is to avoid making wholesale changes that can be unduly disruptive to the workplace culture, while at the same time trying to align the organization with product line goals that cut across project efforts. The working group again returned to the theme of starting small as a key risk mitigation means. Choose a well-scoped product line with modestly scoped organizational change rather than attempt a risky enterprise overhaul.

Providing an Appropriate Funding Model
The funding model is closely linked to the CONOPS, organizational structure, and the business case. This model identifies funding sources to initiate and support the product approach. Developing a suitable funding model involves clearly laying out a product line approach over multiple systems and identifying the life cycle cost savings and benefits to senior level management to obtain their buy-in.

One participant stated that “seed money” is essential to overcoming objections, and without it there may be no practical way to get started and demonstrate savings. Although there was general agreement that the product line startup risk should ideally be addressed through research and development (R&D), the current funding structure often works against this.

Suggestions for creating a funding model include:
- obtaining grassroots support to convince sponsors of the benefit of the product line solution rather than management directing a solution
- reallocating a portion of the funds from programs that will benefit from the product line approach and using those monies to fund the product line
- aligning funding to support the long-term plan and justifying seed money from other areas (including using R&D funds for pilot projects)
- creating a horizontal funding line as a firm part of the budget based on product line feasibility and return on investment

A major barrier cited is that the organizational unit responsible for developing the concept of operations is not usually in charge of the funding model. This re-emphasizes the need for a product line funding mechanism that can align sponsorship with horizontal areas that cut across projects. Other barriers that were discussed include funding instability, parochial views of organizations opposed to the pooling of funds, restrictions on the use of funds (e.g., color of the money), and a lack of incentives for an enterprise approach to systems development that transcends organizational units and commands.

Developing and Implementing an Acquisition Strategy
All of the participants indicated that developing and implementing a suitable acquisition strategy is critical to achieving a product line approach in the DoD. One of the key perceived differences in implementing a product line approach in the DoD environment, as opposed to commercial ventures, is the predominant role acquisition plays. The acquisition strategy defines how to deal with product lines within the contracting environment of the DoD and still be responsive to unique project requirements. One participant suggested that the DoD contracting environment provides a lot of freedom; a big challenge is to find the appropriate contractual vehicle and recognize that the early buy-in and endorsement of the contracting officer and contract negotiator play a pivotal role in the acquisition strategy.

A key issue for the DoD participants in developing a product line acquisition strategy was how to competitively acquire derivative products without endangering contractor interests or the government's ability to maintain control over the core assets. Another concern is the issue of liability for any government-provided components.

A common concern of the group was that proven acquisition approaches (i.e., ones that are repeatable and responsive to life cycle requirements) constitute a major unknown, and will need to be gradually developed, refined, validated in actual practice, and disseminated. Guidance is especially needed on how to include architecture issues in a request for proposals.

The second group of DoD workshop participants identified several specific acquisition strategies. Generally, these strategies differed in the degree to which the government owned the product line assets. In increasing ownership of assets these strategies were:
- to acquire a product built using product line technology (no government ownership of assets)
- to acquire a reference architecture to serve as a basis for future acquisitions of specific system architectures, assets, and products
- to acquire a system architecture and a set of components from which future systems may be built. (The Army Common Hardware/Software system is a successful example of this strategy.)
- to acquire a system architecture, a set of components, and at least one product built using these assets. (The Army Crusader Howitzer program is a successful example of this.)

Generally, as you work up the scale of increasing government ownership of assets, the risks associated with having unvalidated assets decreases. However, the risks associated with the scope of the acquisition, the expense, and the commitment required increases.

Other areas where it was indicated that acquisition guidance is needed to support a product line approach include:
- developing an acquisition plan and selecting a suitable contract vehicle(s) that is compatible with the product line concept and takes full advantage of acquisition reform measures
- preparing solicitation packages and specifying appropriate technical evaluation criteria
- including precautionary measures to minimize the risk of a protest before or after contract award
- incorporating contract incentives to sustain contractor motivation after contract award, and to encourage cooperation and efficiency commensurate with the contractor's role as a product line team player

All of these measures are aimed at overcoming the traditional mindset of a single-system acquisition program and accommodating multiple project efforts.

Contractor Interface
Members of the group observed that at
the organizational level, the interface to
the contractor and the contractor product
line practices seemed to be tightly cou-
ped to the acquisition approach of the
DoD project. At least for traditional, sin-
gle-system acquisitions, the business and
funding models; the organizational struc-
ture and operations; the resource develop-
ment and allocation processes; and other
senior management practices seemed to
be based on the DoD’s customary acqui-
sition practices.

Comparing the traditional enterprise
to the product line enterprise, a few issues
come to the forefront.

The first issue concerns the contrac-
tor’s business model. Contractors now
have multiple business opportunities.
They can focus their business on one or
more of three roles:
- lead contractor for architecture
- subsystem/asset developer
- systems developer/integrator

Having choices raises important ques-
tions, such as:
- What are the criteria that would lead
  a contractor to choose one business
  opportunity over another?
- Would not most contractors opt to
  lead architecture development for the
  contract security and competitive
  advantage it provides over asset
  developers and system integrators?

The second issue concerns shared
commitment. For a product line
approach to be successful, the working
group believed that the contractors and
acquisition organization must share
responsibility and commitment to cost
avoidance through systematic reuse. How
is this achieved?

The third issue concerns contractor
buy-in of a product line architecture.
Systems integrators will not be motivated
to use a mandated product line architec-
ture that may not reflect their design
practices. System development risks and
costs may be greater, particularly if the
contractor has no experience and assur-
ance that the architecture is valid. The
architecture will be “dead on arrival.” How
is this scenario avoided?

Having all interested contractors col-
laborate on developing a product line
architecture may resolve the above issues,
but this may not be feasible in all cases.
For example, the architecture may be an
open systems standard, or only one con-
tractor may have the needed expertise.
In addition, there may be cases when the
performance and schedule risk of an
architecture by consensus is too great.

There are no clear-cut answers, but a
joint government/industry approach to
these issues must be developed for long-
term product line success.

Summary and Conclusions

There are many benefits to a product line
approach and many organizations have
succeeded in realizing these benefits. Yet
there are also costs and risks for any
product line program. Nevertheless, if
properly managed, the benefits of a prod-
uct line approach far exceed the costs.
Strategic software reuse through a well-
managed product line approach holds
great promise for the DoD in terms of
efficiency, time to field mission capability,
and quality.

The SEI vision for product lines is
that this practice will pervade software
engineering in the new millennium, and
we are committed to helping the DoD
succeed in the successful exploitation of
this technology. To assist in this exploita-
tion, the SEI Product Line Systems
Program has established the Business
Acquisition Guidelines project. This proj-
et exists to address product line acquisi-
tion challenges within the DoD. We invite
you to visit our Web site to learn more
about our work in this important area.

The SEI is a Federally Funded
Research and Development Center spon-
sored by the DoD.

About the Author

Lawrence G. Jones is a
senior member of the
technical staff in the
Product Line Systems
program at Carnegie
Mellon University’s SEI.
Before joining the SEI,
Jones served a career in the Air Force and is
the former Chair of the Computer Science
Department at the Air Force Academy.

Jones also serves on the ACM Accreditation
Committee and the Executive Committee
of the Computing Sciences Accreditation
Commission.

References

1. Brownsword, Lisa; Clements, Paul. A
   Case Study in Successful Product Line
   Development (CMU/SEI-96-TR-016, ESC-
   TR-96-016). Pittsburgh, Pa.: Software
   Engineering Institute, Carnegie Mellon
   University, 1996.
2. Bergey, John; et. al. DoD Product
   Line Practice Workshop Report
   (CMU/SEI-98-TR-007, ESC-TR-98-007).
   Pittsburgh, Pa.: Software
   Engineering Institute, Carnegie Mellon
   University, 1998.
3. Clements, Paul; Linda Northrop. A
   Framework for Software Product Line
   Practice, Version 1.1, Pittsburgh, Pa.: Software
   Engineering Institute, Carnegie Mellon
   University, March 1999.
4. Bergey, John; Matthew Fisher;
   Lawrence Jones. The DoD Acquisition
   Environment and Software Product
   Lines (CMU/SEI-99-TN-004, ESC-
   Engineering Institute, Carnegie Mellon
   University, May 1999.
5. Foreman, John. “Product Line Based
   Software Development — Significant
   Results, Future Challenges.” Software
   Technology Conference (STC ’96),
   Salt Lake City, Utah, April 1996.
6. Bergey, John; et. al. Second DoD
   Product Line Practice Workshop Report
   (CMU/SEI-99-TR-015, ESC-TR-99-
   015). Pittsburgh, Pa.: Software
   Engineering Institute, Carnegie Mellon
   University, 1999.
7. Cohen, Sholom. Guidelines for
   Developing a Product Line Concept
   of Operations (CMU/SEI-99-TR-008,
   ESC-TR-99-008). Pittsburgh, Pa.: Software
   Engineering Institute, Carnegie Mellon
   University, 1999.
The Software Insight Tool: A Tool and Methodology for Risk Mitigation and CIO Assessments

Jerry Kastning and Jeff Herman
U.S. Army CECOM Software Engineering Center
Marilyn Ginsberg-Finner and Jim Heil
Telos Corp.

Mitigating Risk and Improving a Program’s Health

The SIT is a device that will guide the user in identifying and addressing software-intensive program strengths, weaknesses, and risk areas. The SIT will greatly aid management in lowering program risks and producing a product with a much greater probability of meeting the customer’s requirements within cost and on schedule. The SIT is designed to become management’s key guide for any risk mitigation program, both on an ongoing basis and to prepare for chief information officer (CIO) assessments and major milestone (MS) reviews. This paper describes the tool, provides background for its development, and describes how it can be used in an internal risk mitigation process, as well as in the CIO assessment process.

The SIT is a practical approach for risk identification when used by a PM or a software integrated product team (IPT) for ongoing or periodic internal risk mitigation reviews or preparing for major program reviews, such as CIO assessments. Using the SIT can identify cost, schedule, and performance risk areas, e.g., why program costs are increasing, why schedules are slipping, and/or where performance and practices are weak. The results will yield a better managed, lower risk program and a product with a much greater probability of meeting the customer’s requirements within cost and on schedule.

Risk Mitigation Through CIO Assessments

CIO assessments are performed to satisfy the requirements of Division E of the Clinger-Cohen Act (formerly the Information Technology Reform Act [ITMRA]) [1], and to comply with the subsequent policy guidance from the Office of the Secretary of Defense (OSD) [2]. At the Department of Defense (DoD) level, its CIO is responsible for

Figure 1. How much can a PM juggle?

SIT provides:
- Essential insight into the health and risk of software programs
- Cost-effective risk mitigation and management of today’s complex acquisitions
- Preparation for required CIO assessments
ensuring that information technology (IT) is acquired and information resources are managed within an integrated management framework, and to assess and manage the risks of DoD’s IT acquisitions (including National Security Systems). Component milestone decision authorities (MDAs) and CIOs will follow similar practices for IT programs subject to their review and approval, and each service was required to provide its implementation of these requirements.

The Army implemented a formal CIO assessment process, which incorporates the Clinger-Cohen and OSD guidance into the Army’s regulatory and acquisition process [3]. The Army CIO is designated to assess Army programs, and recommend to the MDA whether to continue, modify, or terminate the program. The SIT was developed to support the Army implementation, and is used to prepare for the CIO assessment.

An Expert System

The increasing complexities of system acquisition and development, coupled with shrinking resources, require not only extensive knowledge of best practices and streamlined processes, but also expert systems to help assess and satisfy the myriad program and system requirements. The SIT is a knowledge-based instrument that provides a set of questions from which the user selects those of most importance/relevance to the current project status and issues. The SIT does not dictate a set of correct answers or actions. It facilitates identification of program risks and the subsequent planning and implementation of program improvements and risk reduction actions. The SIT is applicable whether development is in-house, by a two-party acquisition/supply agreement, by integration of existing components, by new development, or by any combination thereof.

Concept of the SIT

The SIT presents a comprehensive set of questions to assist acquisition and development management in evaluating a program against statutory and regulatory requirements (e.g. DoD 5000.2-R), as well as software acquisition best practices. DoD 5000.2-R, paragraph 4.3.5, states, “Software shall be managed and engineered using best processes and practices that are known to reduce cost, schedule, and performance risks.” Use of the SIT will help accomplish the DoD 5000.2-R requirements in reducing risk and enhancing software quality [4], as well as reducing TOC. These questions can be used by Army, Air Force, and Navy System Development Offices, and federal government agencies for periodic internal program reviews to reduce software-related risk and in preparation for DoD-mandated CIO assessments or other high-level reviews.

The SIT builds on and complements well-respected sources of best practices and is intended to provide an acquirer-side perspective on plans and practices for acquisition, development, and sustainment. The major sources used for best practices are illustrated in Figure 2 [5, 6, 7, 8, 9, 10]. While there are similarities between program risk mitigation reviews and Capability-Based Assessments-Internal Process Improvement (CBA-IPIs), the target is different. CBA-IPIs are assessments of a developer’s capabilities and maturity based on the CMMI, while CIO assessments and program risk mitigation reviews use the SIT to assess status and risks for the entire acquisition program. The acquirer uses the SIT to assess the acquisition program, rather
Software Acquisition

- Transcending question topics:

1. Overall life cycle approach to the software acquisition and development project.
2. Compatibility with DoD goals and service enterprise-wide objectives.
3. Service-wide and joint interoperability with current and projected systems.
5. Integration of the system into the projected battlefield.
6. Information assurance approach.
7. Overall life cycle software support concept (strategy).
8. Identification of critical program risks; planning for next risk mitigation review.

Figure 4. SIT structure: Transcending question topics.

than just the development effort. The SIT focuses on how the acquirer (the PM and his organization) plans and is progressing in ensuring a well-managed, successful program and the acquisition and support of a system that will meet the needs of the user; the CMMI and the CMM focus on a developer's process capability. The risks identified through using the SIT provide a basis for risk mitigation at all phases of an acquisition program, which may include acquisition and development process improvement. A software capability evaluation (SCE), or a software process risk evaluation (SPRE) performed by the Army, is a CBA that provides a basis for source selection by the acquirer — typically within the engineering, manufacturing/development (EMD) phase of a specific acquisition development project. A developer uses a CBA-IPI to identify development process improvements — typically from an organizational perspective and independent of a specific development project. The SIT can be used on a periodic or ongoing basis, as well as in advance of DoD-mandated CIO assessments. An acquirer relies on a SCE in advance of source selection and may occasionally use it to take a snapshot of an ongoing development process.

The SIT is not a tutorial or handbook on how to plan/manage a project. The SIT questions do not attempt to prescribe the correct way to do things, or prompt the user. The SIT questions are intended to ask how things are actually being done on the project (describe what you are doing) and cause management to focus on the important software/system/program considerations/issues. Most of the SIT questions are not written to yield simple yes/no answers. The questions are open-ended and nondirective, and are designed to obtain descriptive information as a basis for achieving insight into the project status, issues, and risks. The completed responses will be meaningful to management, as well as to life cycle software engineering (LCSE) experts, and should be analyzed to identify any software-related weaknesses and risks in the program.

Structure of the SIT

Two major elements comprise the SIT: a software questionnaire and M atrix+, as illustrated in Figure 3 [11]. The M atrix+ provides an extended version of the basic Army matrix to assess a program against CIO and DoD program requirements. The questionnaire starts with a set of eight high-level transcending questions (TQs), followed by detailed questions in 46 assessment areas.

The SIT Questionnaire — Transcending Questions

There are several questions that are of overriding importance in assessing any program. The TQs are high-level questions that should be reviewed and asked at the beginning of the risk mitigation process and should be used in summarizing key issues, risks, and actions at the end of the process. The TQs do not replace the detailed questions in the assessment areas. However, they are extremely important to the overall success of the system, from a program-wide perspective. Figure 4 provides the TQ topics.

The SIT Questionnaire — Domains and Assessment Areas

The 46 assessment areas are grouped under the seven domains listed in Figure 5. The sequence of domains and assessment areas does not imply a priority. Figure 6 provides sample SIT questions.

All MS reviews in DoD regulation 5000.2-R, “Mandatory Procedures to Major Defense Acquisition Programs (MDAPs) and Major Automated Information System (MAIS) Acquisition Reviews,” were considered [4]. Each assessment area table has columns for each MS (0, I, II, and III), and a column for developmental program reviews labeled PR. The PR column identifies where internal program review considerations should be focused during EMD. If a question is considered relevant for a MS, a bullet is shown in the appropriate MS column(s); if the question is not considered relevant for that MS, then the column is left blank.

Figure 5. SIT structure: The seven domains.

- Seven domains comprise the top level of the SIT Questionnaire:

1. SOFTWARE TECHNOLOGY
2. SOFTWARE ACQUISITION MANAGEMENT
3. PROGRAM MANAGEMENT
4. SOFTWARE PROCESS
5. SOFTWARE QUALITY
6. TEST & EVALUATION
7. SOFTWARE OPERATION & SUPPORT
Matrix+: An Extension to the CIO and DoD Program Requirements Matrix

Based on the Clinger-Cohen Act [1], the OSD provided guidance for its M & S review requirements in the form of a matrix of high-level program requirements that were part of an OSD policy memorandum [2]. The OSD matrix addresses requirements in recent legislative reform initiatives (ITM RA, the Government Performance and Results Act [GPRA] of 1993, and the Paperwork Reduction Act [PRA] of 1995) and related DoD regulations, such as DoD 5000.2-R.

The Army implemented this OSD guidance policy in the Army Policy Memorandum, “Chief Information Officer (CIO) and DoD Program Assessment Requirements,” dated Nov. 14, 1997 [3]. The Army matrix, containing 22 specific CIO and DoD program requirements, was attached. This basic Army matrix was updated in 1998 and is available in Department of the Army (DA) Pamphlet (PAM) 70-3, “Army Acquisition Procedures” (Appendix XIII, “Chief Information Officer Assessment Requirements”) [12] and is contained within the M atrix+ portion of the SIT web pages at www.sed.monmouth.army.mil/sit. With regard to the program requirements in the Army matrix, DA PAM 70-3 (Appendix XIII) states that “Program managers will use these criteria on a continuing basis to evaluate their programs and will incorporate them into their acquisition processes, procedures, and documents.” (The phrase “these criteria” refers to the 22 program requirements found in the Army matrix.) The Army CIO will assess all Army Acquisition Category (ACAT) I and II programs using the Army matrix. All significant ACAT III and IV programs — with information technology expenditures of $2 million or more in a single year, or with a total life cycle cost of $30 million or more — will be evaluated by the appropriate organizations designated responsible for the CIO function at the Major Commands; these programs also will use the criteria in the Army matrix.

The 22 specific program requirements in the Army matrix are at a high level and take an overall program view (see Figure 7 for a list of the 22 program requirements). The Army matrix also has several detailed questions supporting each program requirement, with many of the questions relevant to software issues.

Many of these detailed questions were taken from the SIT questionnaire. The M atrix+, as available in the SIT, is identical in content to the basic Army matrix, except that it provides additional (clearly identified) detailed questions based on selected SIT questions, and provides linked cross-references to the relevant SIT questions and assessment areas. Figure 8 provides sample M atrix+ questions. Note that a “+” in the “milestones” block under MS II and III indicates that there are additional software concerns that also should be addressed, based on some of the additional detailed questions in M atrix+.

**Internal Risk Mitigation Reviews**

The SIT should be used periodically by PMs and their software IPTs to conduct internal reviews of a development/acquisition program to keep the project in good health, reduce the level of program risk, and to be ready for a CIO assessment. Figure 9 depicts the internal SIT risk mitigation process.

Utilizing the SIT for internal risk mitigation reviews on a regular basis will help ensure program success, in that software and program risks will be identified and managed in an ongoing and consistent manner. Internal risk mitigation reviews also will facilitate preparation for the required CIO assessments (the CIO assessments are based on the program requirements in the M atrix+ found in the SIT).

**Figure 6. SIT structure: Sample SIT questions**

**Figure 7. Program requirement areas (M atrix+).**

1. CORE MISSION
2. OUTSOURCING
3. BPR/BENCHMARKING
4. COTS SOLUTIONS
5. RETURN ON INVESTMENT
6. STRATEGIC GOALS
7. TECHNOLOGY
8. YEAR 2000
9. STANDARDS/FLEXIBILITY
10. OPEN SYSTEMS
11. OPERATIONAL TEST and EVALUATION
12. PERFORMANCE MEASUREMENTS
13. FULL FUNDING
14. INCREMENTAL (PHASED STRATEGY)
15. CONTRACT RISK MANAGEMENT
16. COMPETITION
17. EARNED VALUE
18. SOFTWARE SUPPORT ANALYSIS
19. SAFETY, QUALITY, and TESTING
20. SOFTWARE PROCESS IMPROVEMENT
21. INFORMATION ASSURANCE
22. ELECTRONIC COMMERCE

*BUSINESS PROCESS REENGINEERING*
The Data Collection Team

PMs should form a DCT, which includes their software experts and/or software IPT, to adequately respond to the questions. Additional team members should be obtained from the appropriate SSA or LCSEC. To make the reviews meaningful, it is essential that the DCT consists of very knowledgeable, technically qualified software engineering and software acquisition personnel, who thoroughly understand software life cycle issues and are familiar with the project.

Collecting the Data

Reviewing the questions in advance of the data collection should enhance the team’s comprehension, as well as improve the quality and completeness of the answers. The DCT collects the responses to the questions, and copies of certain project material (e.g., software development plans or any other referenced documents or materials). When information already exists in a documented form, the response should reference this information (e.g., citing specific document and paragraph numbers) and copies of the referenced materials should be provided. The answers can be brief where examples and other information are referenced and provided.

Obviously, judgment should be used for selecting and addressing the questions for each project. If a phased approach is to be used, the most critical assessment...
areas should be addressed in the initial phase. For some projects, a question with a bullet in a milestones column may not be relevant, and may be tailored out for a valid reason. If a question or subquestion is not relevant or important to the project, the DCT may tailor it out with a brief, specific justification. Questions should be interphased in a phase-appropriate manner; e.g. if it is too early for an action, the response should describe the plans and approach to be taken (i.e. identify what will be done to ensure that the objective is accomplished).

The Independent Evaluation Team
The IET should be independent of the DCT. IET members should be experienced software, system, and program personnel who understand the technical and programmatic depth and breadth of acquisition and development programs. They also should be trained in the evaluation methodology and understand the goals and activities associated with each assessment area. In addition to identifying program/project strengths, weaknesses, and risk areas, the IET (in coordination with the PM) also can help generate an action plan to rectify the weaknesses and risks.

Transcending Questions
At the end of the SIT data collection and internal evaluations, the SIT user should return to the TQs and summarize the findings (see, particularly, TQ 8 in Figure 4). Risks that have been identified should be included in the project risk management tracking system.

Estimated Time Frames
Depending on the size and complexity of the program's software, a complete internal risk mitigation review (all relevant assessment areas) may require about one month to prepare the answers/responses by the DCT, and approximately two months to evaluate the responses by the IET. Action plan generation will require additional time. Given the review results and the perceived risk, an internal SIT risk mitigation review may be conducted at 10- to 18-month intervals.

Instead of a complete review, a partial review or a series of shorter incremental reviews (e.g. three to six assessment areas per month) may be conducted, each focusing on different assessment areas identified as key for the particular program at its current point in the life cycle. Program risk profiling to identify critical areas for review, or to plan the sequence of incremental reviews, may take one or two days. Another alternative is a mini-review, prescreening using the first question in each relevant assessment area to identify areas of significant risk for further study; the prescreening would take about two weeks.

Protecting PM Information
Protection must be provided to the responses, and the findings should be given only to the PM. The PM can ask the IET for specific recommendations to address any weaknesses or risks found, and may also ask the IET for support in the preparation of an action plan to address the weaknesses and risks.

CIO Assessments
Purpose of the CIO Assessment
The CIO assessment will be conducted prior to M S reviews, consistent with the Clinger-Cohen Act (ITMRA) [1] and the related DoD policy [2]. In the Army, these assessments are based on the program requirements in the Army matrix [11, 12]. Other services and DoD agencies may utilize similar CIO assessment processes to ensure that programs meet the DoD and service information technology program requirements [1, 2]. Throughout the CIO assessment, it should be foremost in the minds of the various teams and the PM staff that the CIO assessment is intended to support the PM in ensuring successful acquisition of high-quality, supportable systems and software to meet the critical needs of DoD war-fighting personnel.

CIO Assessment Overview
A program preparing for a CIO assessment should utilize the SIT risk mitigation process as the front-end of the CIO assessment to help ensure the success of the CIO assessment and the related major M S review. Figure 10 illustrates the use of the SIT for CIO assessments.

The upper portion of Figure 10 identifies the PM data collection process for the CIO assessment, which utilizes the SIT risk mitigation process. The lower portion of Figure 10 identifies the additional activities in the Army CIO's assessment process. A CIO assessment evaluation team (AET) evaluates the data the PM submitted and makes program recommendations to the CIO. Then the CIO makes the recommendation to the Defense Acquisition Board or appropriate MDA to modify, continue, or terminate
the program.

Preparing for a CIO Assessment

Preparing for a CIO assessment should start several months prior to the M S reviews specified in D O D 5000.2-R. A significant time saving will be realized where internal risk mitigation reviews have been performed previously and regularly. A sufficient amount of time should be allowed (e.g. about one month each) for the D C T to prepare the answers and for the I E T to analyze the answers and generate findings. The I E T will analyze the responses to the Army matrix (or M atrix+) questions, as well as to appropriate S I T questions. (To be better prepared, the PM should address the additional questions in the M atrix+ and selected questions from the assessment areas.) The I E T will identify any weaknesses, potential problems, or risks, and discuss them with the PM. The PM thus will be informed of potential risks in advance of the CIO assessment. The PM should formulate an action plan, with help from the I E T, to address any weaknesses, problems, and risks. Work should begin prior to the CIO assessment and major M S review, on the actions to proactively address these issues and reduce project risk.

About two to three weeks should be allowed for the PM and the I E T to revisit the questions in the CIO assessment (for the Army, the basic Army matrix¹) and prepare the final answers to be sent to the CIO. The PM should have actions already under way to address any issues before sending the response to the CIO. The PM can proactively develop an action plan (with help from the I E T) prior to, or concurrent with, submission of the responses to the CIO.

The CIO Assessment Evaluation

The CIO AET will then analyze the responses to the Army matrix questions to determine strengths, weaknesses, and any significant risks, and report its findings/recommendations to the CIO. The AET will need about one month to complete the evaluation and report the findings to the CIO. The findings and recommendations should be completed and made available well before the formal M S review to allow time for the CIO to review and, if necessary, discuss any concerns or issues with the PM. The final assessment result will be the CIO’s decision to recommend continuation, termination, or modification of the program to the M S decision authority (M DA). The decision of the M DA M S Review is then fed back to the PM. The PM can request recommendations from the AET.

Protecting CIO Information

The D C T, the I E T, and the CIO AET must protect the information and treat the findings as sensitive information to be given only to the individual who chartered the team's effort, i.e. the PM and/or the CIO, or their designated representatives.

Summary

The purpose of the S I T is to support program management (i.e. the acquirer) in identifying and addressing software-intensive program strengths, weaknesses, and performance risks to meet critical needs of the soldier, the airman, or the sailor, and to reduce overall program risk and T O C. The focus of either an internal risk mitigation review or a CIO assessment is on identifying potential or actual performance problems and risks, on identifying potential areas for cost or schedule overruns, and on giving the PM advance opportunity for resolution or mitigation of problems and/or risks. The S I T will help the PM keep the project on the road to success and to be prepared for CIO assessments.

S I T Access

The S I T is a Web-based tool and may be accessed from the CECOM M SEC Web page, www.sed.monmouth.army.mil/sit.

It may also be accessed from (1) the Army DISC 4 Web page, www.army.mil/disc4/acq. (Scroll down to "Software Development and Engineering Insight" and select "Software Insight Tool to Prepare for Milestone Reviews"); and (2) the D O D Under Secretary of Defense Acquisition and Technology (USD [A& T]) Director, Test, Systems Engineering, and Evaluation (D T S E & E) risk management Web page, www.acq.osd.mil/te/programs/se/risk_man

About the Authors

Jerry Kastning is the Division Chief for the Strategic Planning and Policy Division of the Operations Directorate for the Software Engineering Center (SEC) at Fort Monmouth, N. J.

Kastning has been with SEC for the past 14 years. In addition, he has held several first line supervisory positions in SEC, including the Acquisition Streamlining and Risk Management, Information and Engineering Support Branch, the Automatic Test and Software Support Branch, the Software Configuration Management Branch and the Office of Process Improvement. Prior to his employment with SEC, he was the software project leader with PM T M E D (Test Measurement and Diagnostic Equipment) for eight years. In this capacity he was responsible for the development and testing of all system and application software. He also has many years of hands-on experience with designing, coding, and testing various software application programs for military systems, as well as computer hardware design.

Kastning earned his bachelor of science and master's of science degrees in electrical engineering from the University of Illinois and a master's degree in business administration from Monmouth University.

Division Chief
Software Engineering Center
AMSEL-SE-O PS-SPPD

Jerry Kastning
Division Chief
Software Engineering Center
AMSEL-SE-O PS-SPPD
James Heil is a technical manager at Telos Corp., Fort Monmouth, supporting the CECOM Software Engineering Center. Previously, he worked at MITRE — Fort Monmouth Software Engineering Center, on Fire Control Systems software, Acquisition Support, and software Standards (IEEE/EIA-1498) and ISO 12207.

Earlier, Heil was a software development manager and SQA manager at ITT Avionics, and a software development manager at IBM.

He was CODSIA Industry Task Force chairman for MIL-STD-2167 and 2167A, and served on MIL-STD-498, IEEE 1498/EIA 640 (J-016) and 12207 working groups.

He chaired the NSIA SQA group, and has led conferences and seminars on software quality, process improvement, software acquisition, standards, and testing. He chairs the North Jersey SPIN. Heil was an invited participant at Orlando II and San Antonio DoD Workshops.

He authored “Practical Applications of Software Quality Assurance to Mission-Critical ... Software” in Handbook of Software Quality Assurance, 2nd Ed., and 3rd Ed.

Heil has a bachelor of science degree in mechanical engineering and a master’s degree in electrical engineering, and a master’s degree in industrial engineering (operations research), and a master’s degree in business administration.

References
1. “Clinger-Cohen Act of 1996 (Division E),” also referred to as the Information Technology Management Reform Act (ITMRA) of 1996.

Note
1. For the Army CIO assessment, only the responses to the basic Army matrix questions are required; however, responses to the “plus” questions provide assurance that critical software issues are covered and are a basis for action by the PM, where needed. Further, using the assessment areas can identify relevant issues at M Ss that are not specifically noted as applicable in the Army matrix. The responses to the basic Army matrix will be reported to the CIO in advance of the M S review and analyzed by the CIO assessment evaluation team. If the responses do not provide adequate information, follow-up information may be needed. Using the more detailed questions in the SIT will facilitate more thorough analysis in advance and can expedite the response.

AcqNOW! is a subscription service that provides immediate e-mail notification of changing acquisition information. Simply fill in the electronic subscription form at http://www.deskbook.osd.mil/acqNOW and mark the information types that affect you when they change: FAR, DFARS, DoD D5000.1 and 5000.2-R, SAM M, DCAAM, and others.

As changes become available, AcqNOW! will send a brief e-mail and a link to the “raw” source document on the Deskbook Web site.

If you are already a subscriber but have changed your job or address, click AcqNOW! and update your contact information, along with any changes in your information preferences.

Acquisition reform is accelerating; AcqNOW! to keep up.
Software and the 15th Century

I received an interesting e-mail looking at the origins of some of our modern sayings that derived from life in the 15th century. Although I question its accuracy, I could not help but see a parallel with today’s software engineering. Here are my observations:

**Marriage and Funding:** In the 15th century most people took their yearly bath in May and got married in June because they were still smelling good in June. To be safe, brides carried a bouquet of flowers to hide their body odor.

Today, software projects are started in September to avoid the loss of fiscal year surplus funds. To be safe, project managers carry around justification statements to avoid the smell of working the system.

**Baths and Personal Computers:** In the 15th century, baths consisted of a big tub filled with hot water. The man of the house had the privilege of the nice clean water, followed by the other men, the women, and finally the babies. By the time the babies had their turn, the water was so dirty you could lose someone in it. Hence the saying, “Don’t throw the baby out with the bath water.”

Today, executives enjoy the privilege of the best personal computers and peripherals, followed by the Chief Information Officer and his/her staff, other managers, and finally software engineers. By then the computers available to the engineers are so slow and archaic you could lose someone to a competitor with better equipment. Thus a new saying, “Don’t throw the engineer out with the 486.”

**Roofs and Defects:** In the 15th century, houses had thatched roofs, with decaying straw piled high and no wood underneath. In the winter it was the only place for dogs and cats to get warm. When it rained it became slippery and sometimes the animals would fall off the roof. Hence the saying, “It’s raining cats and dogs.”

Today, software programs are hastily thrown together. Spaghetti code is piled high with no structure underneath, providing a hospitable place for Ghastly Non-Architectural Traps (GNATs) and Big Ugly Goofs (BUGs) to germinate. Thus a new saying during software testing, “It’s raining gnats and bugs.”

**Death and Peer Reviews:** In the 15th century, lead cups were used to drink wine. The combination of lead and alcohol would sometimes knock people out for a couple of days. Someone walking along the road would take the person for dead. The unconscious person was laid out on the kitchen table for several days and the family would gather around, eat and drink, and wait to see if their loved one would wake up. Hence the custom of holding a “wake.”

Today, software engineers sometimes mix Mountain Dew and chocolate bars, causing them to bounce off their cubicles like a battling top. Colleagues walking by and recognizing the potential for spontaneous combustion take the engineer into a conference room. Other engineers are invited to help talk the whirling engineer down to earth. Hence the custom of “peer reviews.”

**Graves and Networks:** In the 15th century, England started running out of places to bury people so they would dig up coffins and reuse the grave. In reopening these coffins, one out of 25 coffins had scratch marks on the inside, indicating people were being buried alive. They decided to tie a string on the deceased’s wrist, lead it through the coffin and up through the ground and tie it to a bell. Someone would sit through the graveyard all night to listen for the bell. Hence on the “graveyard shift” they would know that someone was “saved by the bell” or he was a “dead ringer.”

Today, software engineers are buried in massive cubical pits and overloaded with tasks. To measure productivity, management strings fiber optics from each cubicle through a router to a massive network. The engineer is asked to communicate with peers through e-mail and prohibited from Internet use. Hence on the “7/24/365 shift” managers know which engineers are “saved by the e-mail” and which are “web ringers.”

The more things change, the more they stay the same.

— Gary Petersen, TRI-COR Industries

Got an idea for BACKTALK? Send an e-mail to backtalk@stsc1.hill.af.mil
Acquisition Software Oversight
The Deputy Under Secretary of Defense gives her thoughts on software acquisition.

Using Contractor Capability Evaluations to Reduce Software Development Risk
Two methods used during source selection to evaluate a contractor's software development capability.

Effective Acquirer/Supplier Software Document Reviews
Reviewing project plans, requirements, and test plans for software acquisition success.

Product Line Acquisition in the DoD: The Promises, The Challenges
Results of two Department of Defense product line workshops.

The Software Insight Tool: A Tool and Methodology for Risk Mitigation and CIO Assessments
Lowering program risks and producing a quality product with the Software Insight Tool.

From the Publisher: Integrating Acquisition with Software and Systems Engineering

The Data and Analysis Center for Software Announces a New Technical Report

BackTALK: Software and the 15th Century

Coming Events
An Open Invitation to Subscribe to AcqNOW!

Departments and Announcements