Instead of the Wrecking Ball

Rudy Alder
Publisher

Software development is in trouble. A 1990 study of software acquisition in one large U.S. federal government organization showed that only 1.5 percent of major software projects were used as delivered, 3 percent were used after modification, 19 percent were used but later abandoned within two years, 29 percent of the software was never delivered to contract, and 47.5 percent was delivered but never used.[1] In commercial industry, the average software project overshoots its schedule by 50 percent, 33 percent of the projects are canceled and of those that are not, 75 percent are declared operational failures.[2] By all rights, the entire industry should be condemned as unsafe, bulldozed over, and rebuilt from scratch.

Of course that would be impractical. Whatever reforms are made have to originate from within the current framework. Acquisition reform is one such effort to improve those abysmal statistics. Unfortunately, the way acquisition reforms are being implemented usually falls far short of the intended purpose.

The problem is twofold: First, recent work-force reductions leave many government programs few options but to outsource its acquisition functions to contractors. Although many contractors possess superior acquisition skills, fewer government people for oversight functions means a greater probability of miscommunication and subsequent project failure.

Second, the government has decades of experience in acquiring hardware but only a few years in the acquisition of software. When acquiring a tank, the government merely draws up the specifications and waits for the finished product to roll off the assembly line. It is then relatively easy to test the tank's capabilities against the specifications: It either shoots straight or it does not.

But hardware acquisition cannot be used as a model for software acquisition because software is fundamentally different from hardware; software is distilled human intelligence, a collection of abstract ideas buried in the bowels of silicon memory. It makes no contact with human senses except through a hardware interface. This inherent intangibility means that greater care must be used to specify, regulate, and test the product. Unlike the hardware acquisition process, the user must be involved in every step of software's creation as it occurs. "Black-box" testing at the end of the process will not suffice—a "white-box" review of the software during development also is essential to yield a better understanding of what is being created. With fewer government employees to oversee the development of the desired product, prospects for meeting user requirements dim considerably.

Acquisition reform to this point has not been based on a recognition of these essential differences between software and hardware; until it is, most reforms will serve only to further perpetuate the poor performance of most software development projects.

Until that situation is remedied, those who wish to improve their acquisition capabilities will have to rely on other fixes. The Software Technology Support Center is equipped to assist projects in various aspects of acquisition management such as managing expectations, defining processes, or just-in-time skills. Our consultants can assess your current status and suggest improvements that could save your project from the wrecking ball. Contact us by phone, fax, E-mail, or surface mail; the addresses and numbers are on the inside back cover and in the center insert. ◆

Notes

1. *Systems Testing and Quality Assurance Techniques: Fundamentals*, Vol. 1, Ver. 4.1S, January 1997, p.VI-34.
2. *Guidelines for Successful Acquisition and Management of Software-Intensive Systems*, Software Technology Support Center, Hill Air Force Base, Utah, Ver. 2.0, June 1996, p.1-10.

Have Trainees Verify Uses for New Skills

I will disregard the common perception that management is merely giving lip service to training, particularly in these days of severe budget restraints, but I have some comments on Paula Shafer's article, "Planning an Effective Training Program," *March 1998*.

With respect to "management support" and motivation, I recall seeing a training film here on base (Hill Air Force Base) years ago in which a female clerk was sent to several training programs to upgrade her skills. After she completed the programs, in the same breath that her boss congratulated her, he asked her to make another pot of coffee, i.e., business as usual. Then he was surprised when she wanted to move to another job. I am waiting to see people being moved to jobs where their acquired skills can be more fully used. Now, that would be motivation.

Shafer gave several valuable suggestions under "Theory vs. Practice" with respect to using the skills acquired. I suggest another: A patented training program of the Boy Scouts of America, "Woodbadge," uses the unique (in my experience) device of having the trainees specify in writing how they will use the skills learned "on the job" over the next six-month to two-year period, and commit to doing it. Successful completion of the course is withheld until trainees submit a written report detailing their experiences in using those skills.

Tim Layton
Operational Flight Program Support, Avionics Software Test Section

EXECUTIVE ORDER

- - - - - - -

YEAR 2000 CONVERSION

The American people expect reliable service from their government and deserve the confidence that critical government functions dependent on electronic systems will be performed accurately and in a timely manner. Because of a design feature in many electronic systems, a large number of activities in the public and private sectors could be at risk beginning in the year 2000. Some computer systems and other electronic devices will misinterpret the year "00" as 1900, rather than 2000. Unless appropriate action is taken, this flaw, known as the "Y2K problem," can cause systems that support those functions to compute erroneously or simply not run. Minimizing the Y2K problem will require a major technological and managerial effort, and it is critical that the United States government do its part in addressing this challenge.

Accordingly, by the authority vested in me as President by the Constitution and the laws of the United States of America, it is hereby ordered as follows:

Section 1. Policy. (a) It shall be the policy of the executive branch that agencies shall:

(1) assure that no critical federal program experiences disruption because of the Y2K problem;

(2) assist and cooperate with State, local, and tribal governments to address the Y2K problem where those governments depend on federal information or information technology or the federal government is dependent on those governments to perform critical missions;

(3) cooperate with the private sector operators of critical national and local systems, including the banking and financial system, the telecommunications system, the public health system, the transportation system, and the electric power generation system, in addressing the Y2K problem; and

(4) communicate with their foreign counterparts to raise awareness of and generate cooperative international arrangements to address the Y2K problem.

(b) As used in this order, "agency" and "agencies" refer to federal agencies that are not in the judicial or legislative branches.

Sec. 2. Year 2000 Conversion Council. There is hereby established the President's Council on Year 2000 Conversion (the "Council").

(a) The Council shall be led by a chair who shall be an assistant to the president, and it shall be composed of one representative from each of the executive departments and from such other federal agencies as may be determined by the chair of the council (the "chair").

(b) The chair shall appoint a vice chair and assign other responsibilities for operations of the council as he or she deems necessary.

(c) The chair shall oversee the activities of agencies to assure that their systems operate smoothly through the year 2000, act as chief spokesperson on this issue for the executive branch in national and international fora, provide policy coordination of executive branch activities with State, local, and tribal governments on the Y2K problem, and promote appropriate federal roles with respect to private sector activities in this area.

(d) The chair and the Director of the Office of Management and Budget shall report jointly at least quarterly to me on the progress of agencies in addressing the Y2K problem.

(e) The chair shall identify such resources from agencies as the chair deems necessary for the implementation of the policies set out in this order, consistent with applicable law.

Sec. 3. Responsibilities of agency heads. (a) The head of each agency shall:

(1) assure that efforts to address the Y2K problem receive the highest priority attention in the agency and that the policies established in this order are carried out; and

(2) cooperate to the fullest extent with the chair by making available such information, support, and assistance, including personnel, as the chair may request to support the accomplishment of the tasks assigned herein, consistent with applicable law.

(b) The heads of executive departments and the agencies designated by the chair under section 2(a) of this order shall identify a responsible official to represent the head of the executive department or agency on the council with sufficient authority and experience to commit agency resources to address the Y2K problem.

Sec. 4. Responsibilities of Interagency and Executive Office Councils. Interagency councils and councils within the Executive Office of the President, including the President's Management Council, the Chief Information Officers Council, the Chief Financial Officers Council, the President's Council on Integrity and Efficiency, the Executive Council on Integrity and Efficiency, the National Science and Technology Council, the National Performance Review, the National Economic Council, the Domestic Policy Council, and the National Security Council shall provide assistance and support to the Chair upon the Chair's request.

Sec. 5. Judicial Review. This Executive Order is intended only to improve the internal management of the executive branch and does not create any right or benefit, substantive or procedural, enforceable at law or equity by a party against the United States, its agencies, or instrumentalities, its officers or employees, or any other person.

WILLIAM J. CLINTON

THE WHITE HOUSE
February 4, 1998

Capability Maturity Model Process Improvement

Mark D. Schaeffer
Office of the Under Secretary of Defense

The Software Engineering Institute (SEI) is a federally funded Research and Development Center with the mission to accelerate the most effective technology and practice of modern software engineering. The SEI is funded primarily by the Department of Defense (DoD) but also accepts work from other government organizations as well as the private sector via Cooperative Research and Development Agreements.

The centerpiece product of the SEI has been the Software Capability Maturity Model (CMM) released in 1991. This model has contributed to widespread success in assisting organizations in improving their efficiency in developing quality software products. The success of the Software (SW) CMM spawned other CMMs that address a wide range of subjects.

A CMM provides an organization a conceptual framework within which specific processes, e.g., configuration management and quality, can be optimized to efficiently improve the capability of organizations. A CMM provides state-of-the-art practices to

- Determine the maturity of an organization's processes.
- Establish goals for process improvement.
- Set priorities for immediate process improvement actions.
- Plan for a culture of product or service excellence.

By focusing on specific processes, an organization can best leverage the resources for their improvement activities while rallying the organization around specific goals. A CMM can be a road map showing an organization how it can systematically move to more mature levels of performance and do it in more effective and efficient ways. After an objective assessment, an organization can set its goals for increasing the capability of its processes. To the DoD, this translates into more affordable products and services for our war fighters.

CMMs can include processes that span the entire lifecycle. Starting with requirements management, they can span the breadth of product development, ensuring quality, lean production concepts, and support to the field. Each individual process includes elements that provide basic practices as well as additional practices that add incremental benefits and maturity. When these processes are sufficiently matured, the organization increases its performance or maturity.

Subsequent to the success of the SW-CMM, other CMMs were developed with SEI support. These CMMs included the Systems Engineering CMM and the Integrated Product Development (IPD) CMM. It became apparent in the development of these and other models that they all contained common processes, e.g., configuration management, quality, and requirements management, supporting the various functional disciplines, software engineering, and systems engineering. Improvements in these common processes could benefit other disciplines. Further, it became apparent that process improvement resources applied to one functional discipline, e.g., software engineering, could be beneficial to another functional discipline. The common elements used in a software CMM appraisal could be used for a systems engineering appraisal, and there would be no need to redo the appraisal of common elements. In addition, improvement efforts based on unique CMMs could result in suboptimization, confusion, and potentially unnecessary expenditure of process improvement resources.

Acquisition reform in the DoD created a significant paradigm shift away from a "how-to" mentality approach to an approach centered on Statements of Objectives and Performance-Based Requirements. The earlier capability models and standards were clearly used in the context of meeting contract requirements. There were even brief attempts to use them as selection criteria or as compliance benchmarks rather than frameworks to identify and define characteristics of good practices that facilitate process improvement. Remember the Requests for Proposals that required an SW-CMM Level 2 or above to propose? Although DoD Directive 5000 directs we select capable suppliers, it does not direct how it should be determined or set arbitrary levels. DoD has learned over time two important things about maturity levels:

- Many organizations have benefited from the use of CMMs as process improvement tools resulting in delivery of improved products to DoD and government.
- Many projects or products delivered by organizations, purported to be at the SEI Level II or Level III, have not met the customers' requirements.

One of the top-priority projects in the SEI is integration of the CMM products for use in single or multiple functional disciplines. Industry and government along with the SEI now have enough experience in the various functional disciplines to build this framework upon which all present and future CMMs can be based. This will greatly enhance the efforts of CMM users and protect the resources already invested. Organizations can use their previous CMM process improvement work and tailor their future efforts to their unique organization. The initial common framework effort will be based on the SW-CMM, the SE-CMM, and the IPD-CMM. Other functional disci-

Editor's note: As of press time, this article represents the most recent direction for Department of Defense CMM-related efforts. CROSSTALK will publish more information on the CMM Integration effort as it becomes available.

plines may be added later. To efficiently use the government funds allocated to CMMs, further work on CMMs that are not common framework compliant has been halted. The work accomplished to date in Software CMM, Version 2.0 and the IPD CMM have been included in the initial CMM Integration (CMMI) baseline.

In building these CMMI products, the needs of industry and government partners must be understood and met. We have had extensive participation in our reviews of the CMMI requirements, and broad collaborative efforts are underway developing the products. We are depending on the functional discipline experts from industry and government to assist in building the products.

In summary, the CMMI project requires a broad collaborative effort to ensure that the best practices are included and process improvement resources are optimized. Industry along with government and the SEI are participating on a team to build the CMMI products. Since many organizations have already made considerable investments in CMM-oriented process improvement efforts, it is important that the products of this project efficiently integrate into these efforts, and that resources are not wasted on a new approach. ◆

About the Author

Mark D. Schaeffer has over 20 years experience in weapons systems acquisition and program management in the Office of the Secretary of Defense, Naval Sea Systems Command, and as congressional staff. He has been the deputy director for systems engineering since November 1994 and is responsible for policy and implementation of systems engineering, technical risk management, design for manufacturing quality, reliability and maintainability, manufacturing, and acquisition logistics.

Coming Events

Software Quality Through Robust Testing

Dates: May 21, 1998
Location: Eatontown, N.J.
Subject: Gain confidence in year 2000 fixes, reduce testing time and cost, improve coverage, and find defects early.
Contact: Madhav Phadke, Voice: 732-577-2878; Fax: 732-577-2879; E-mail: Madhav_Phadke@compuserve.com

7th IEEE North Atlantic Test Workshop

Dates: May 28-29, 1998
Location: West Greenwich, R.I.
Subject: Issues for the 21st Century: higher quality, more economical, and more efficient testing methodologies and designs.
Sponsor: IEEE Computer Society, Test Technology Technical Committee, University of Rhode Island
Contact: Jim Monzel, Voice: 802-769-6428; Fax: 802-769-7509, E-mail: jmonzel@vnet.ibm.com

Effective Methods of Defect Detection and Defect Prevention

Dates: June 2-4, 1998
Location: Seattle, Wash.
Subject: "Software Quality," decomposed into defect detection and defect prevention.
Sponsor: Quality Assurance Institute
Contact: Voice: 407-363-1111; Fax: 407-363-1112; Internet: http://www.qaiusa.com

5th International Conference on Software Reuse

Dates: June 2-5, 1998
Location: Victoria, British Columbia
Sponsor: IEEE Computer Society in cooperation with Association for Computing Machinery
Contact: Dr. Jeffrey S. Poulin, program co-chairman; Voice: 607-751-6899; Fax: 607-751-6025; E-mail: Jeffrey.Poulin@lmco.com

Second Workshop on Software Architectures in Product Line Acquisitions

Dates: June 8-10, 1998
Location: Hawthorne Hotel, Salem, Mass.
Subject: Applying software architecture technology to acquisition of all or parts of a line of software-intensive systems. Based on government and industry experiences, working groups will make recommendations for moving to an architecture-based acquisition approach for a product line.
Contact: Lt. Col. Gene Glasser, E-mail: glassere@issc.belvoir.army.mil

15th International Conference on Testing Computer Software

Dates: June 8-12, 1998
Location: Washington, D.C.
Subject: "Testing Under Pressure," with emphasis on management strategies.
Sponsor: U.S. Professional Development Institute
Contact: Voice: 301-270-1033; Fax: 301-270-1040; E-mail: admin@uspdi.org; Internet: http://www.uspdi.org

Software Cost and Schedule Estimation Course

Dates: July 13-15, 1998
Location: University of California at Los Angeles
Subject: Many issues associated with project cost and schedule estimation, why projects succeed or fail, advantages and disadvantages of widely used models, year 2000 challenge, emerging issues, and reference sources.
Sponsor: UCLA Extension Short Course Program
Contact: Marcus Hennessy, Voice: 310-825-1047; Fax: 310-206-2815; E-mail: mhenness@unex.ucla.edu

Slaying the Software Dragon

Lt. Col. L. John Michel, III
Information Resources Management College, National Defense University

It continues to be a daunting task to educate members of the Department of Defense acquisition work force who acquire, develop, engineer, test and evaluate, conduct research on, or procure software-intensive systems. The Information Resources Management College and the Defense Systems Management College have developed a software acquisition management curriculum that meets the certification requirements and educational needs of the community. It takes "dragon slayers" armed with the knowledge of software acquisition management to slay the "beast."

"640K ought to be enough for anybody." – Bill Gates, circa 1981

The beast continues to grow. Our systems are becoming more software intensive because software is replacing the functionality formerly performed by people and hardware. Rear Adm. Robert M. Moore, former commander of the Naval Information Systems Management Center, identified this transition in March 1993, when he stated,

> "At one time, it was the hardware that supported the mission. Today, the hardware is rather generic, capable of supporting any mission. It is the software that provides the real functionality."

In a 1992 report to the House Armed Services Committee, the General Accounting Office (GAO) estimated that total annual software cost would account for 20 percent of the Department of Defense's (DoD) budget by 2008 [1]. Last October, Federal Sources, Inc. completed a survey of defense spending on software used for weapons systems, information systems, and command, control, communications, computers, and intelligence systems [2], excluding software for nontactical systems. The report projected that by 2002, DoD will spend over $20 billion annually on software.

Software acquisition and development within DoD continues to be a significant management problem. Software is the critical component of today's defense systems. A variety of studies and analyses over the past 13 years have continued to identify significant systemic software acquisition problems. The beast is fed by a dearth of software acquisition management education.

The Dragon's Lair

People who are not masters of software technology and acquisition management build the dragon's lair. Many studies relate DoD's "software crisis" to a need for software acquisition management education. In September 1987, the Defense Science Board Report on Military Software, office of the under secretary of defense for acquisition, stated,

> "Application-knowledgeable, technically skilled leaders are the military's *limiting resource* in acquiring today's computer technology. … Few program offices are staffed [properly] due to a shortage of qualified people. ... [T]he DoD should implement the education and training necessary for its *software acquisition management personnel* to master both software technology and *acquisition management*."

The DoD Software Master Plan, Vol. I (draft), February 1990, developed by the Defense Acquisition Board Science and Technology Committee, reported,

> "Improving software education and training is critical. ... [T]here is a need to coordinate efforts of the National Defense University, Defense Systems Management College, and Industrial College of the Armed Forces to integrate software *acquisition* and development programs into existing courses and to establish mandatory software engineering education for all DoD technical and contractual personnel involved in the *acquisition* process."

The DoD Information Systems (IS) Work Force Education, Training and Career Development, Executive Resources Task Force Report (October 1992) asserted,

> "Technical vitality of the IS work force is critical to effectively deploy information systems in support of the DoD war-fighting mission. The need to provide recurring technical training to individuals, *especially at midcareer and executive levels*, was communicated throughout our meeting with services, agencies, and private industry. This training is essential to ... keep pace with the acquisition of more advanced computer and telecommunications systems."

The Crusade

> "The educated differ from the uneducated as much as the living from the dead." – Aristotle

The "crusade" is not a "death march." There is a critical need for a work force highly trained in the complex programmatic discipline of software acquisition management. The need for a review of the DoD's software acquisition management education and training curricula and career programs was identified in May 1993 by the Acquisition Management Functional Board, an organization that advises DoD component executives in the management of accession training and career development of acquisition work force personnel. On Oct. 19,

Course	Level	Target Audience	Format	Prerequisite
SAM 101	I	GS-9 and below and military ranks 01-03	Distance education, 18 lesson modules.	ACQ 101
SAM 201	II	GS-9 and above and military ranks 03-04	10-day classroom-based curricula.	ACQ 201, SAM 101
SAM 301	III	GS-12 and military ranks 04 and above	10-day classroom-based curricula.	SAM 201
NDU Elective 5546		GS-13 and military ranks 05 and above	Fall and spring semester, two hours weekly for 12 weeks. Seminar format.	

Table 1. *Course offerings. SAM 101 is expected to be available in October 1998.*

1993, the Terms of Reference for the review of software acquisition management education was approved by Colleen A. Preston, then deputy under secretary of defense for acquisition reform.

Approved in March 1994, the team's report established a set of nine critical competencies and 24 key competency areas. In addition, the report contained the recommendation for the development of assignment-specific mandatory courses for software acquisition personnel for Level I, Level II, and Level III training career levels.

Arming the Dragon Slayers

> "Technology is dominated by two types of people: those who understand what they do not manage, and those who manage what they do not understand."
>
> – Anonymous

The dragon slayers must be armed with the education to manage the acquisition and development of the growing number of software-intensive systems—to annihilate the dragon. Under the auspices of the Defense Acquisition University (DAU), the Information Resources Management College (IRMC) at the National Defense University (NDU), and the Defense Systems Management College (DSMC) began joint development of an evolutionary course curriculum. IRMC was tasked to lead the design of the capstone course, Software Acquisition Management (SAM) 301, and DSMC led the design of the basic and intermediate courses, SAM 101 and SAM 201.

The software acquisition management courses are assignment-specific. These courses have been identified by the under secretary of defense for acquisition and technology as integral to the education of acquisition work force personnel. They are a means to provide unique acquisition knowledge required for a specific assignment, job, or position. They maintain proficiency while remaining current with legislation, regulation, and policy. The SAM courses are for people who acquire, develop, engineer, test, evaluate, conduct research on, and procure software-intensive systems.

For students attending the colleges of the NDU, the National War College, the Industrial College of the Armed Forces, and the IRMC's Advanced Management Program, software acquisition management education is available through the NDU electives program. Future Directions in Software Management (Elective 5546) investigates cutting-edge practices for developing high-quality software-intensive systems. This course focuses on challenges that face program management personnel, managers of software development organizations, information management officers, and corporate information officers involved in the acquisition of software-intensive systems. The course shows how to manage software acquisitions using state-of-the-practice methods and techniques and lets the operator and acquirer gain a mutual perspective on the issues involved in acquiring systems that sustain the war fighter.

The Challenge

> "Sometime they'll give a war and nobody will come."
>
> – Carl Sandburg

Few dragon slayers have taken up their swords and joined the crusade to slaughter the dragon. During the 1997 academic year, the two colleges were prepared to educate 325 students in 13 SAM 301 offerings. Reality was 66 DoD students in eight offerings, and the colleges had to combine courses to have class sizes that facilitated the seminar format. For SAM 201, the numbers have been equally dismal.

Why the low response? Maybe it is because software acquisition is not perceived to be a career field. Maybe it is because the realization has not sunk in that all systems are information systems and the ubiquitous thing that moves, manages, manipulates, and presents that information is software. Maybe it is the drawdown of the acquisition work force. Maybe it is because SAM courses are another set of education requirements that takes the person out of the workplace. For whatever reason, the "software education crisis" is not being rectified, and the dragon still roams the land, largely unchallenged by properly equipped dragon slayers.

The students' evaluations of the courses indicate that the colleges have developed quality programs that meet the needs of the software acquisition professional. This shortfall means that seats are readily available. Join the crusade to slay the software dragon.

Join the Crusade

Join the small legion of dragon slayers—make the choice and grow. The SAM 301 and SAM 201 schedule for the remainder of 1998 is as follows:

SAM 301

April 27-May 8, DSMC, Fort Belvoir, Va.
June15-26, IRMC, Fort McNair, Washington, D.C.
Aug. 17-28, DSMC, Fort Belvoir

SAM 201

June 15-25 IRMC, Fort McNair

Learn About the Crusade

"Education's purpose is to replace an empty mind with an open one."
– Malcolm Forbes

For more information on these courses, visit the following Web sites or contact the faculty.

http://www.ndu.edu
http://www.dsmc.dsm.mil
http://www.acq.osd.mil/dau

SAM 301
Lt. Col. L. John Michel, III
Voice: 202-685-2062 DSN 325-2062
E-mail: Michell@ndu.edu;
Larry Baker
Voice: 703-805-3636 DSN 655-3636
E-mail: bakerl@dsmc.dsm.mil

SAM 201
Lt. Col. Rob Simmons
Voice: 703-805-5419 DSN 655-5419
E-mail: simmons_rob@dsmc.dsm.mil
Dr. Michael Martin
Voice: 202-685-4880 DSN 325-4880
E-mail: martinm@ndu.edu

SAM 101
George Prosnik
Voice: 703-805-3578 DSN 655-3578
E-mail: prosnikg@dsmc.dsm.mil ◆

References

1. General Accounting Office Report, IMTEC-92-62BR.
2. Seffers, George I., "Survey: Software Cost to Exceed Hardware," *Defense News*, Nov. 10-16, 1997, p. 6.

About the Author

Lt. Col. L. John Michel, III is a professor of systems management at the IRMC at the NDU. He is course manager for SAM 301 and Future Directions in Software Management. His fields are software acquisition, interoperability, and architectures. He has over 16 years experience in the development of command and control, intelligence, and combat support systems.

Information Resources Management College
National Defense University
300 Fifth Avenue
Fort McNair, D.C. 20319-5066
Voice: 202-685-2062 DSN 325-2062
Fax: 202-685-3974 DSN 325-2062
E-mail: Michell@ndu.edu
Internet: http://www.ndu.edu/

Software Capability Evaluation Reuse and Reform

Industry had complained for years that it was swamped by government demands for software capability evaluation-related paperwork and visits, and different agencies often wanted the same information. The situation became common enough to acquire a name: redundant reviews.

"Industry continues to assert that it seems as if the government is constantly looking at them, asking the same questions and getting the same answers," said Lt. Col. Charles F. Vondra, U.S. Army aquisition reform staff officer, office of the deputy under secretary of defense. "This is a great example of what acquisition reform was meant to fix. A typical Software Capability Evaluation (SCE) costs the government an estimated $50,000, and contractors say it costs them a similar amount. When an SCE repeats an earlier evaluation, it just wastes time and money. The overall goals of the entire process are to make consistent, reliable information widely available, to save money, and to ensure equitable treatment of contractors."

Last July, R. Noel Longuemare, acting under secretary of defense for acquisition and technology, directed the Systems Engineering Steering Group to find ways to improve the system for performing SCEs.

The current policy is to reuse the results of earlier evaluations whenever possible by updating them so that they reflect an offeror's current capability. A system being implemented to assist in executing this policy consists of

- The software center operated by the Defense Contract Management Command (DCMC) in Boston, Mass. will collect all information on source-selection SCEs conducted on Department of Defense (DoD) contractors.
- Completed SCEs will reside at the Air Force Electronic Systems Center, a DoD SCE repository established at Hanscom Air Force Base, Mass. This repository draws on work by the Army's Research and Development Engineering Center, Communications-Electronics Command and the Naval Command, Control, and Ocean Surveillance Center.
- All government-sponsored teams (not just DoD teams) will have access to the SCE results.
- Information will be safeguarded as source-selection sensitive.
- Results of SCEs will be shared with offerors, who may provide comments that will also be put into the repository at Hanscom.

As in so many other acquisition reform initiatives, a hero of the SCE story is an Integrated Product Team. After Longuemare's directive, the "SCE Team" was established to find ways to use SCE information more systematically.

As it happened, the team was able to build on earlier work. A tri-service group was formed four years ago to promote the use of SCEs in evaluating development risk and to improve consistency in applying the SCE method.

Then, last year, a pilot program was launched at the Electronic Systems Center to reuse SCEs, which resulted in approximately $1 million cost avoidance. This pilot led to the formation of the Government SCE Consortium last May, also led by the Air Force's Electronic System Center. The consortium's future role will be to provide a forum for sharing experiences and to gather new ideas to improve the application of SCEs and reusing them.

"The Acquisition Reform office is also looking at government-performed SCEs as temporary," Vondra said. "We would eventually like to see a commonly accepted evaluation by an independent third party, similar to ISO 9000 quality certification in manufacturing."

"We want a professional, consistent system with reliable information that is less intrusive to industry and uses information smarter and faster." ◆

This announcement is based on "SCE Reuse: Ending Redundant Reviews," AR Today, January/February 1998, Vol. 3, No. 1.

Maximum-Leverage SCE Techniques

Richard T. Bechtold
pragma Systems Corporation

This article describes a set of high-leverage Software Capability Evaluation (SCE) techniques that facilitate reuse of contractor-provided SCE data and that shift objective process maturity determination from an acquisition agency to contractors performing work for that agency. This approach can contribute to a substantial reduction in the number of SCEs an agency performs while ensuring contractor compliance with required process maturity levels. This article is of potential interest to any government agency performing SCEs and to any company that is currently or potentially subject to contractually based Capability Maturity Model requirements.

As with having nuclear weapons, the most successful strategy for SCEs is to never have to use them. If an acquisition agency or a prime contractor organization desires, for example, Capability Maturity Model [1] (CMM) Level 3 contractors to perform prime or subcontract work, it would be ideal to know confidently, without having to perform SCEs, the true maturity level of contractors submitting proposals. Additionally, during contract monitoring, it would be ideal if an agency could be confident that their contractors are continuing to maintain the required maturity level without the agency having to repeatedly perform SCEs.

Depending on the number of software contracts an agency monitors, the time and effort to perform SCEs on each contractor can become prohibitive. For example, consider the challenge if the Defense Contract Management Command (DCMC) wanted to ensure the maturity on the software-intensive contracts it oversees. Since the DCMC currently has 6,600 software-intensive contracts [2], it would appear nearly impossible to perform regular SCEs on each contractor.

This article describes a set of high-leverage SCE techniques that not only facilitate reusing contractor-provided SCE data but also shift responsibility for objective process maturity determination from an acquisition agency to the contractor performing work for that agency. This approach can potentially contribute to a substantial reduction in the number of SCEs an agency must perform to ensure contractor compliance with required software process maturity levels.

The premise of these techniques is that SCEs can be performed by an agency in a manner that, over time, encourages contractors to provide results from SCEs performed by other agencies. These techniques also encourage contractors to objectively self-appraise and self-report detailed appraisal information. This can minimize, for the agency and the contractor, the cost and effort associated with a government agency determining and monitoring process maturity. Simultaneously, this approach helps contractors have the clearest picture of where to focus their process improvement efforts.

Maximum-Leverage SCE Techniques

A high-leverage SCE technique is any technique that, when performed in support of an SCE, substantially improves the quantity or quality of SCE information or substantially reduces the effort required to gather it. Each of this article's high-leverage techniques is valuable in isolation, but maximum leverage is best achieved by using a majority of these techniques in combination. The techniques described toward the end of this article-which share the common characteristic of reusing existing appraisal data-are especially high leverage.

Select High-Content Projects

A high-content project is any project that can provide usable evidence across a significant number of key practices (presuming the project is above Level 1). In principle, a Level 2 project can readily provide evidence across nearly all of the Level 2 key practices. However, in practice, an SCE team may find that some projects-through no fault of their own-do not map well to the CMM.

The key is to find projects in which contractors own the processes they are following. In some environments, especially major government environments such as the Department of Defense (DoD) and the Federal Aviation Administration (FAA), it is common to find certain processes that are written, owned, and mandated by the acquisition agency. Therefore, it is almost impossible to gain insights into a contractor's configuration management practices, for example, if that contractor is contractually obligated to follow highly explicit and detailed configuration management procedures mandated by the acquisition agency. High-content projects are those in which the contractors own, and are responsible for, the processes they are following.

Additionally, do not too quickly exempt a project from an SCE simply because it does not involve writing code-it may still be a high-content project. Project managers might assert that their teams are not doing software development because they are not writing "IF" statements and "WHILE" loops. The project might involve designing a database schema or developing a requirements specification. However, from the perspective of ensuring the successful engineering of software-intensive systems, any contractor that owns the processes it follows and materially contrib-

utes to the success or failure of the specification, design, development, maintenance, or migration of a software-intensive system, can be a source of valuable insights into organizational software process maturity.

Ensure Some Projects Have Usable Subcontract Evidence

Subcontract management is sometimes outside the scope of an SCE team because the subcontract vehicle does not allow ready comparison to the CMM. Two common examples are subcontracts used to acquire the expertise of consultants and subcontracts used to provide temporary staff augmentation to an existing project team.

Projects may need consultants to provide expert opinions, advice, or specialized experience not available from existing organizational resources. These subcontractors are not expected to perform software planning, tracking, and oversight activity as reflected by the CMM Subcontract Management key process area (KPA) [3]. It is not uncommon for the Subcontract Management KPA to have little application to this type of consulting arrangement, even though such arrangements are sometimes put in place using subcontracts.

The second type of subcontract arrangement involves augmenting an existing project team with outside people who have additional or internally unavailable skills. These arrangements are characterized by essentially identical treatment of the subcontractors and the prime contractor's team. In such an environment, an outsider may find it difficult to determine who is employed by the prime team and who by the subcontractor. Again, this is not "traditional" subcontracting, and such projects should generally not be considered a good source of insight into the ability of a prime contractor to perform CMM-compliant subcontract management.

Cover Seven to 10 Projects

Many SCEs include only three or four projects in their evaluations. The risk of this approach is that if one or two projects prove to be difficult to compare to the CMM, the entire SCE becomes difficult to complete.

Any contractor that claims significant accomplishments and the ability to perform software engineering projects should be able to offer a variety of projects for SCE review. Start with the objective of reviewing at least seven to 10 projects, which increases the likelihood of having at least four or five projects that can be readily compared and contrasted to the CMM.

To accomplish this, it is advantageous to first examine project profiles for 12 to 15 projects. First, it encourages a larger number of the contractor's project teams to consider whether they are performing at the required maturity level. Even the teams that are not selected for the SCE, having come so close, might thereafter work harder at improving their processes. Second, this improves the likelihood of finding the best high-content projects. And third, by starting with a larger pool of projects you can better accommodate contractor preferences with regard to selecting projects where the SCE will not adversely affect deliverable deadlines or critical milestones.

Pre-Qualify Interviewees

Pre-qualification starts with an analysis of contractor-provided project organization charts, which typically include names, titles, and a general depiction of management, reporting, and command relationships. You should tentatively identify approximately twice as many people as you intend to interview and ask the contractor to provide one-paragraph descriptions of the work performed by these people.

Upon receiving these descriptions, you should eliminate approximately one-fourth of the candidates, then request résumés for the remainder. Upon receiving and reviewing the résumés, you should eliminate approximately one-third of the candidates, which leaves you with a pre-qualified group of your intended size.

As you review and down-select the candidates, be sure to get a mix of all types of employees. You will want veterans and new hires, highly experienced and novice workers, and generalists and specialists. Such diversity will typically result in comprehensive, complete, and accurate data regarding organizational process maturity.

Pre-Plan Extra Interviews

This step provides several advantages. First, it allows you and the SCE team to truly follow the evidence. As people describe their work, you can ask questions designed to elicit names ("Who else reviews your work?" or "Who from quality assurance helped you with this?) If a name is not on your current interview list but the person seems like a good source of information regarding organizational processes, you can insert that person in one of the open interview slots. Also, when an interviewee misses an assigned time due to sickness, a project crisis, or any other valid reason, it is easy to reassign that person's interview to an open slot.

Try to leave at least two open interview slots during each day of the on-site period. Also, have extra open slots scheduled for the last day or two of the SCE; this allows you time to identify additional interviewees. Be sure to fill at least half of these open slots with extra interviews.

Avoid Functional Area Representative Group Interviews

When assessing a project to initiate process improvement, simultaneously interviewing a group of eight to 12 people is an excellent way to obtain a wealth of information about its processes in a particular functional area. More important, it helps bond these people into a group and helps increase personal commitment and buy-in from the participants.

However, when conducting an SCE, functional area representative group interviews are much less effective. One or more participants may be perceived by others in the group as not completely trustworthy with regard to nondisclosure or confidentiality agreements. Participants will also be much less inclined to say anything useful, because anything they say will be heard by the group and

may therefore reach management. Subdued interviews lead to a bad irony for the contractor, because SCEs are generally a "proof-positive" exercise. That is, the SCE team should already be familiar with the contractor's documented processes, but if the team encounters no incidental evidence to verify that the contractors are truly working in a CMM-compliant manner, it may evaluate one or more KPAs as not fully satisfied.

It is usually to the contractor's advantage during the SCE to have an atmosphere where people feel they can openly discuss the work they perform. Having individuals meet privately with the SCE team usually best provides that atmosphere.

Double-Interview the Project Managers

It is difficult to know how to schedule project manager interviews. If you interview the project managers first (a fairly common practice), after listening to several days of technical and functional area interviews, you often wish you could bring the project managers back and ask a variety of more specific questions. Conversely, if you interview the project managers last, you often wish you could bring back a number of the technical people to further explore, verify, and validate the statements made by the project managers.

Although currently an uncommon practice, you will likely find it useful to interview project managers both at the beginning of the on-site week (usually immediately following the executive interviews) and again at the end of the on-site period. For the initial interviews, follow the standard practice of asking prepared questions of a general nature. Then, during the week, build specific detailed questions as a function of the information you hear, or fail to hear, during the technical interviews. Use these specific questions during the follow-up interviews to gain additional insights into project process capability.

Never Go Completely Outside

It is often useful to use the services of one or more external vendors or government organizations that specialize in the performance of SCEs. One advantage to using these outside SCE resources is the high likelihood that they have more experience performing SCEs than your team. By augmenting the team with external resources, your SCE team members will likely become far smarter far faster than they otherwise might have.

However, there is a definite disadvantage to using only outside resources. Your team likely understands the details of your acquisition better than anyone outside the agency. These insights are an important factor in understanding the context in which a contractor's processes are being used. Furthermore, by providing a stable core of resources to perform SCEs, you leave yourself the option of switching between external vendors or using multiple external vendors simultaneously while still ensuring that the SCE approach used by your organization is consistent, and all of the contractors are treated equally.

Never Cancel the On-site

Many SCE teams now perform the majority of the document review prior to the on-site period. This can lead to the inclination to cancel the on-site period if there are clear and significant inadequacies within the submitted documents. For example, you might be reviewing for compliance with Level 2 and find no evidence of policies, procedures, or plans for requirements management, quality assurance, and configuration management. Since it would seem clear that the contractor is not performing at Level 2, it would seem logical to consider canceling the on-site period.

In reality, it will usually make more sense to continue with the SCE. First, there is the remote chance that the documentation you need does exist, but the contractor was too unfamiliar with the SCE process to know it should have been sent to you. During the on-site period, you may hear people repeatedly refer to material that you have not reviewed, which may contain the necessary evidence of Level 2 compliance. Second, the premise behind this set of high-leverage SCE techniques is to motivate the contractor to perform process improvement and self-appraisals in such a way that you rarely need to perform SCEs on that contractor. Therefore, once you commence with an SCE, you should perform the on-site period to provide the contractor with the most complete and comprehensive picture of their process maturity as reflected by their documentation and the activities performed.

In the case of a blatantly noncompliant contractor, you might want to replan and reduce the time spent during the on-site period.

Mutual-Aid SCE Resources

Fire departments and rescue squads routinely use mutual aid as a means to help, and be helped by, their neighboring communities in times of need. Each group maintains the approximate number of people needed for its typical workload, then assists other groups in times of crisis. Mutual-aid agreements anticipate future needs and are executed in a manner that is mutually beneficial to all involved groups.

With regard to performing SCEs, let us assume that you can perform current or routine contract monitoring with X number of SCE employees. A new acquisition, however, might require you to have 1.5X, 3X, or 4X SCE employees available to perform all the necessary SCEs in a timely manner (usually by performing simultaneous or overlapping SCEs).

Receiving SCE resources from another agency to augment your SCE team not only provides you with a surge-mode capability but also facilitates an increased exchange of SCE experiences and lessons learned. To help another agency, or a different area within your agency, by providing them with SCE employees allows your employees to become experienced more rapidly than they otherwise would.

Reuse Appraisal Data

As alluded to in some of the previous techniques, enough companies have been performing self-appraisals and have been evaluated by government agencies that there is now the possibility that you can avoid performing an SCE by reusing

relevant, recent, objective, and convincing appraisal data.

It is generally acknowledged that SCEs are more qualitative than quantitative. Consequently, during and after the performance of an SCE, the SCE team often finds itself having to deal with the issue of confidence levels. For example, if some members of the team are "extremely" confident that a contractor is Level 2, some are "highly" confident, and one team member is "fairly" confident, the team will typically come to consensus that the contractor is Level 2.

Given the above, a critical question for a specific acquisition is, how confident do you need to be that a contractor will perform at or above a given maturity level? To whatever degree you do not have to be "completely" confident, there is an increasing likelihood that you can reuse data from SCEs by other agencies and reuse data from contractor self-appraisals (remembering that self-appraisals can range from highly subjective to fairly objective). By carefully analyzing this data for objectivity, timeliness, relevancy, and consistency, a review team can become sufficiently confident that a contractor is at a certain maturity level, and there is no current need to perform an SCE on that contractor.

Refresh Appraisal Data

When asking contractors to submit details about SCEs and self-appraisals that have occurred within their organization within the last 12 to 24 months, you will sometimes find that the data submitted is somewhat convincing yet still somewhat doubtful. In essence, you need more data. One option is to perform an SCE. If a contractor has virtually no reusable appraisal data, this certainly makes sense. However, if the contractor had a considerable amount of reusable appraisal data, but it was not quite convincing, you might need to refresh the appraisal data by asking for additional information.

The additional information you should request is entirely consistent with the data for which you would ask during an SCE. However, at this point you can ask for far less information since you are not yet performing an SCE. For example, you might ask three or four projects to submit current documentation covering three KPAs within the CMM. This documentation should include policies, procedures, plans, status and tracking reports, etc.

Review of this data will yield one of three results. The review team

- will be convinced that the contractor has the necessary level of maturity.
- could not find convincing data and therefore recommends proceeding with a formal SCE.
- thinks that a brief on-site meeting with the contractor may provide the final necessary evidence. Only in this last instance will you need to use the following techniques.

Augment Appraisal Data

Certain data is typically included as part of a standard SCE process. This data, as mentioned above, includes policies, procedures, plans, guidelines, status reports, etc. At times you will have (somewhat) reusable appraisal data and (somewhat) refreshed project data, but the review team will still be unable to come to consensus regarding a contractor's software process maturity.

One option is to just give up and perform a new SCE. However, you may also be able to augment existing data with a series of briefings with the contractor. These briefings could help the review team better understand, for example, the relevance of a contractor's other Level 3 divisions as opposed to the division that will perform the work you require.

By carefully preparing a list of requested briefings, each of which addresses specific areas where you need more information, combined with the information you already have, you may achieve sufficient confidence that a contractor has achieved the maturity level needed for your acquisition without having to perform an SCE.

Give Considerable Lead Time to Contractors

There are instances when contractors have been given as little as three or four weeks notice that a government agency is coming on site to perform an SCE. This essentially forces the contractor to focus on "successful SCE techniques" vs. "successful process improvement." Ironically, this situation causes some SCE teams to suspect they are witnessing more act than reality, leading to a lose-lose situation for everyone. But if contractors have three to six months to prepare, some will spend that time becoming higher maturity organizations-a win-win situation for all involved.

Share Detailed Evaluation Results with the Contractor

Sometimes, contractors receive little feedback on the results of their SCE. In extreme cases, they only learn whether they won the contract. In such instances, the SCE may have been worthwhile in the agency's search for a contractor, but it will have virtually no value in helping the contractor know where to focus efforts on CMM-based process improvements. Although it would be presumptuous for an SCE team to offer advice on how a contractor should improve its processes, it is in everyone's best interest for the SCE team to share its impressions of the contractor's strengths and weaknesses within various KPAs.

Summary and Conclusions

There are essentially four "golden principles" that govern the application of maximum leverage SCE techniques:

- If you have to perform an SCE, perform it in a manner that maximizes the likelihood of producing reusable SCE data.
- After an SCE, always provide detailed information to the contractor so that the contractor has the option of supplying that information to other agencies.
- Prior to an SCE, always request any pre-existing SCE and appraisal data.
- Aggressively strive to avoid performing unnecessary SCEs.

By following the techniques described in this article, you increase the likelihood that your agency and other agencies will be able to reuse the data from any SCE you perform. This data, when combined with other SCE data and data from contractor self-assessments, can be systematically analyzed

and may allow your agency to validate a contractor's assertion of a maturity level without having to perform yet another SCE on that contractor. (For details on a systematic, multiphase approach for validating contractor process maturity through reused, refreshed, and augmented appraisal data, see [4].)

It continues to be true that the SCE method is the most reliable approach available to evaluate software processes against the software CMM. However, the ongoing accumulation of data from agency- and contractor-conducted SCEs increases the probability that your agency can validate the maturity level of a contractor without requiring your agency (and the contractor) to invest the considerable time and expense required to prepare for and perform an SCE. Central to this idea is when you perform an SCE, you do so in a manner that facilitates potential reuse of the SCE data, and you provide the contractor with the detailed results of that SCE.

By performing maximum-leverage SCE techniques, you can expand the number of contractors being monitored for process maturity and expand the frequency of your monitoring without necessarily having to increase the resources needed to perform the monitoring. Additionally, by regularly and effectively monitoring contractors more closely, you can help prevent the occurrence of a contractor slipping from a higher maturity level to a lower one-an adverse situation that both you and the contractor would prefer to avoid.

Although an obvious objective of these techniques is the performance of highly successful SCEs, the most important objective is to support the performance of successful acquisitions and do so in a manner that recognizes, facilitates, and rewards successful contractor process improvement endeavors. ?

About the Author

Richard T. Bechtold is vice president for product development at pragma Systems Corporation. He was previously a research associate and software project management professor at George Mason University, where he spent most of his time in research, grant, and contract work. His major areas of work include process improvement, business process reengineering, CMM-compliant process evaluations and appraisals, high-fidelity process modeling, process definition, computer-based training, and distance and collaborative learning. He has 18 years experience in software engineering and software project management. He spent seven years at the Software Productivity Consortium, has held a variety of software program and project management positions, and participated in the specification, design, and development of numerous software-intensive systems. He holds a doctorate in information technology from George Mason University.

pragma Systems Corporation
8704 Lee Highway, Suite 303
Fairfax, VA 22031
Voice: 703-560-4669
E-mail: rbechtold@pragmasystems.com

References

1. Paulk, M.C., et al., Capability Maturity Model for Software, CMU/SEI-93-TR-24, Carnegie Mellon University, Software Engineering Institute, Pittsburgh, 1993.
2. Holt, Kevin E., "Software Acquisition Support in the Defense Contract Management Command," CROSSTALK, Software Technology Support Center, Hill Air Force Base, Utah, March 1997, p. 25.
3. Paulk, M.C., et al., Capability Maturity Model for Software, CMU/SEI-93-TR-24, Carnegie Mellon University, Software Engineering Institute, Pittsburgh, 1993, Section L2, p. 44.
4. Bechtold, R., "Efficient Software Maturity Tracking," Proceedings of the Ninth Annual Software Technology Conference, Salt Lake City, Utah, April 1997.

Adding Product Lines, Architectures, and Software Reuse to the Software Acquisition Capability Maturity Model

Tara Ragan, *U.S. Army Space and Missile Defense Command*
Donald J. Reifer, *Reifer Consultants, Inc.*

Under sponsorship of the Ballistic Missile Defense Organization's Small Business Innovative Research program, we have been actively developing and promoting changes to the Software Acquisition Capability Maturity Model that stimulate increased software reuse through revised and improved acquisition practices. Such practices are aimed at helping buyers incorporate product lines, architectures, and reuse considerations into their decision processes and products throughout the acquisition lifecycle. Such practices are not only aimed at improving the way government program offices do business, they are also directed at enhancing the manner in which contractors manage their suppliers, especially in acquisitions that involve commercial-off-the-shelf packages and strategic partnerships.

A paradigm shift has occurred in the way systems are being built. As more systems are developed to open system standards, products once constructed from scratch using custom designs are being replaced by product lines that contain large quantities of commercial-off-the-shelf (COTS) hardware and software components. Commercial and aerospace firms, as part of their current best practices, have adopted these changes in the way systems are being built. For example, Lucent has modified its software process to review all projects before they are given go-ahead to ensure they conform to their architectural standards. Aerospace firms like Northrop Grumman have done the same for product lines they have developed, promoting software reuse across weapons systems within the radar and air defense application domains.

As the emphasis has shifted to product lines and architectures, considerable changes have been occurring in the area of software process. The Software Engineering Institute (SEI) has been changing the frameworks many organizations use to assess the maturity of their software processes to include product line, architecture, and software reuse concepts. For example, the SEI is strongly considering including a new key process area (KPA) at Level 4 of its Software Capability Maturity Model (SW-CMM) [1] called "Organization Software Asset Commonality." This KPA requires organizations to exploit commonality that exists among software products using state-of-the-art techniques like domain engineering.

To exploit commonality, this KPA requires organizations to identify the software product lines that constitute their core business and to populate these with reusable assets when appropriate. Other changes are also being incorporated into Levels 2 and 3 KPAs in the forthcoming Version 2 of the SW-CMM to support reuse. For example, the Level 2 KPA on subcontractor management is being broadened to encompass improved acquisition management and software supplier management practices. These improvements focus on improving the manner in which relationships with suppliers (subcontractors, strategic partners, COTS package vendors, etc.) are managed.

Unfortunately, the Software Acquisition Capability Maturity Model (SA-CMM) [2] has not kept pace with the advances of the SW-CMM in the area of product lines, architectures, and software reuse. The SA-CMM is a sister framework to the SW-CMM that can be used by organizations that purchase development of their software from third parties to assess the maturity of processes used for software acquisition management. Such organizations include government program offices and commercial firms that contract for their software, e.g., many banks and insurance firms outsource their software to third parties, or buy it via strategic partnerships, e.g., the way firms like Boeing buy software for their commercial aircraft. Most of the process community would argue that these types of software acquisition organizations should be at the same level of maturity as those software organizations they are trying to manage.

This article summarizes the results of a Ballistic Missile Defense Organization (BMDO)-sponsored Phase I effort conducted to determine what changes to the SA-CMM are needed to exploit the advances being made in the areas of product lines, architectures, and software reuse. Our Phase I effort recommended over 30 changes to the SA-CMM [3] and confirmed that there is a market for aligned products and services [4]. It also validated the promise of these changes using pilot project appraisals and ensured that these proposals are consistent with Version 2 of the SW-CMM. During Phase II, we will prototype these products and services and demonstrate their value by continued beta testing on trial projects.

The SA-CMM Framework

The SA-CMM describes the processes software buyers use to acquire, sustain,

and maintain software. It provides a framework to establish benchmarks and improve an organization's software acquisition processes. It is a staged model in which processes are organized into KPAs with an architecture similar to the SW-CMM. There are five levels of process maturity through which software acquisition organizations evolve:

- Level 1: The Initial Level – the organization does not have documented processes. It functions ad hoc and relies on crisis management techniques to address problems.
- Level 2: The Repeatable Level – the organization fosters discipline through basic practices, which are followed at the project level.
- Level 3: The Defined Level – acquisition practices are defined at the organization level and are tailored for use at the project level.
- Level 4: The Quantitative Level – metrics-based practices are used to make decisions as processes are employed throughout the organization.
- Level 5: The Optimizing Level – continual improvements are made to processes based on quantitative feedback flowing from early adopter projects.

Figure 1 illustrates the structure of the SA-CMM. It was primarily developed to help government program offices improve the way they manage organizations that develop software for them under contract. However, the use of the SA-CMM is not confined to situations where software is being acquired in this manner. It can also be used in commercial applications where software is subcontracted, outsourced, or acquired from vendors (COTS packages, tools, etc.).

Recommended Changes to the Framework

As part of our Phase I efforts, we performed a detailed analysis of the SA-CMM to scope the changes needed to take advantage of product line, architecture, and reuse concepts. Our goal was to determine what changes were needed to stimulate adoption of software reuse concepts as part of the processes program offices use to manage the acquisition of their software. Definitions for key terms, desired outcomes, and best practices were those previously agreed to by Department of Defense (DoD) representatives, published in 1995 as the DoD Software Reuse Initiative's Strategic Plan [5].

As a result of this analysis, 34 changes were recommended to the existing SA-CMM framework. To develop our recommendations, each of the framework's KPAs was analyzed, along with related change requests submitted to the SEI during the past year. Based on discussions with the SEI principals, our suggested changes were structured as examples, elaborations, and extensions to existing material to minimize the impact of the changes on the overall document.

We did not believe the existing framework needed to be altered. Instead, we opted to provide its users with guidance on how to make the existing framework work for reuse with product lines and architectures in mind. For example, we suggested adding reuse considerations to acquisition plans and recommended that owners of the architecture be delineated in the software acquisition plan, with their roles and responsibilities.

Level	Focus	Key Process Areas
5: Optimizing	Continuous Process Improvement	Acquisition Innovation Management, Continuous Process Implementation
4: Quantitative	Quantitative Management	Quantitative Acquisition Management, Quantitative Process Management
3: Defined	Process Standardization	Training Program, Acquisition Risk Management, Contract Performance Management, Project Performance Management, Process Definition and Maintenance
2: Repeatable	Basic Acquisition Management	Transition to Support, Evaluation, Contract Tracking and Oversight, Project Management, Requirements Development Management, Solicitation, Software Acquisition Planning
1: Initial	Competent People and Heroics	

Figure 1. *SA-CMM key process areas.*

Validation by Experts

The proposed changes were peer-reviewed by a group of software reuse experts from government and industry at the Reuse '97 Workshop held in Morgantown, W. Va. in July 1997. We wanted to ensure that the software reuse and process communities agreed with our changes. After reaching consensus, we submitted the changes to the SEI for incorporation into the next release of the SA-CMM. The experts who participated in the peer review included

- Mary Beth Chrisis, SEI (guest)
- Dixie Garr, Texas Instruments
- Ted Lewiston, U.S. Air Force
- Fred Maymir-Ducharme, Ph.D., Lockheed Martin
- Stan McVay, West Virginia University (scribe)
- Mark Paulk, SEI (guest)
- Sabrina Raman, Boeing
- Donald Reifer, Reifer Consultants, Inc. (facilitator)
- Ken Song, Department of National Defense, Canada

This peer review exercise was extremely valuable because the group endorsed the changes we proposed to the SA-CMM without reservation. They also suggested additional changes to the SW-CMM that were aimed at helping the SEI guests overcome some problems they were having in tackling software reuse issues within the then current version.

Piloting the Results

We also wanted to ensure that our suggested changes stimulated increased reuse. We sought out organizations within government and industry that would let us prove the value of our ideas via what we called a "quick-look" appraisal of their programs. It is interesting to note that few of the program offices we approached were interested in conducting a complete, formal appraisal. They were either too busy or did not have the staff to support the formal appraisal process. They also were concerned about the increased workload and scrutiny that could result from the appraisal's findings. As such, appraisals turned out to be hard to sell.

We began the piloting task by modifying the SEI-developed SA-CMM appraisal questionnaire to incorporate product

line, architecture, and software reuse considerations. We soon concluded that it would be easier to replace the SEI-developed questionnaire with a new one because software reuse concepts could not easily be retrofitted into the document. In response to this need, we developed our own appraisal questionnaire [6]. We also developed a briefing that provided an overview of our Small Business Innovative Research (SBIR) effort and used it several times to solicit acquisition organization participation in pilot appraisals.

Using our questionnaire, we conducted a series of "quick-look" appraisals of the acquisition processes used by two Northrop Grumman and two U.S. Army program management offices. These appraisals validated that the changes recommended to the SA-CMM are valuable and stimulate increased reuse. They had the added benefit of providing the pilot projects with constructive improvement recommendations in areas other than software reuse. They also proved useful in helping us identify priorities for candidate products and services we will provide during our Phase II effort to quicken the transfer of reuse technology via acquisition process alignment.

The lessons we learned as we tried to get projects to participate in appraisals helped us understand the issues software acquisition managers have relative to the SA-CMM and reuse. Their concerns are summarized in the following five questions:

- Why should I conduct an SA-CMM appraisal? What are the costs and benefits?
- What can I do with the results of the appraisal? Where is the leverage?
- Why should I be concerned with product lines, architectures, and software reuse?
- What can I do to improve my management of COTS products? What processes make sense, and what can I do about them? What about enterprise-wide licensing?
- How do I improve the way I manage my strategic partnerships with my suppliers?

Responses to these questions, which appear in our final report [7], were instrumental in getting program office agreement to participate in an appraisal. The most pervasive of these questions dealt with COTS management. Most organizations we talked to had experienced difficulties with COTS and were looking for ways to improve the practices used to manage its acquisition. For example, integrating COTS into the architecture using "glueware" sometimes seemed to create more problems than were solved by the use of COTS.

Another set of concerns stemmed from the fact that most offices we assessed had ongoing programs that manage existing contractor or supplier relationships. People in these offices were interested in knowing the answer to the question "How do I improve my acquisition management processes on contracts that have already been awarded?" Because they were comfortable with their existing practices, it was difficult to sell them on the need to insert new, reuse-based processes.

Market Survey

In parallel with these activities, we conducted a market analysis and developed a business plan and business case [8] to excite support among potential investors for our Phase II SBIR activities. We began this activity by scoping the market for prospective SA-CMM products and services. We wanted to ensure that whatever we proposed to develop during our Phase II SBIR effort had high commercialization potential. We developed a market survey questionnaire and used it to canvass targeted organizations, both industry and government, to determine the size and characteristics of the market for prospective SA-CMM products and services. The results of the survey were extremely encouraging. They indicated that the annual return on our projected Phase II investment should be close to 58 percent. These high returns justified our plans to pursue Phase II support.

We then used the results of the survey to develop a business plan and business case. This plan was used to show investors the potential returns if they elected to fund our future activities. Based on the positive results we projected in this plan and during the pilot appraisals, Northrop Grumman elected to partner with us to solicit Phase I Interim and Phase II fast-track funding from our BMDO sponsors. Their cash investments have been instrumental in helping us secure BMDO Phase I Interim and Phase II funding.

Potential Phase II Products and Services

Based on the results of our market survey and our piloting efforts, we can conclude that a market seems to exist for the following SA-CMM products and services:

- Model Software Acquisition Processes – Model software acquisition management processes that respond to user requirements need to be developed for each identified market segment. Specifically, they are needed for the Acquisition Planning, Solicitation, and Evaluation SA-CMM KPAs. Model processes are also needed in COTS management; even though COTS management is not specifically addressed in the SA-CMM, it was the area where the need for additional guidance seemed most pressing during our piloting efforts.
- Tailoring Guidelines – Related tailoring and scaling guidelines are needed so organizations can apply the model processes within their operations. Guidelines should be aimed at acquisition management, supplier management, and COTS management audiences.
- Software Acquisition Education and Training – A variety of course materials are required to sell executives on the need to use the SA-CMM. Development of practitioner skills, knowledge, and abilities in the model processes and tailoring guidelines also seemed to be desired.
- Appraisals – Appraisals need to be conducted to identify organizational strengths and weaknesses relative to the requirements of the SA-CMM. Organizations may also need help developing improvement plans that respond to the appraisals findings.

- PC-Based Tools – A hypertext-based set of tools that run on a PC are needed to help users tailor Reifer Consultants, Inc. (RCI)-developed model processes to their needs using the tailoring guidelines developed for that purpose. This tool kit should make it easier to use the processes than to not use them.
- Technology Transfer Kits – Everything needs to be packaged so a potential user of the SA-CMM can quickly develop the know-how to use the technology. The preparation of CD-ROMs with everything a user needs to use the processes, in kit form, is being pursued.
- Consulting – Clients may need help using the products and services listed above to assess and develop their software acquisition management process improvement plans.

Our plan is to develop and pilot most of these products and services as part of our Phase II SBIR effort. Close coordination and cooperation with the SEI, BMDO, and our Phase II partners is essential. To keep the effort synchronized and keep all the key players involved, we plan to form an advisory council that will meet regularly to provide us with oversight, direction, and guidance. This council will be chaired by U.S. Army Space and Missile Defense Command, the BMDO-designated manager of this SBIR effort. Members of the council will be drawn from government, industry, and the SEI.

Organizations interested in participating in our Phase II SBIR effort are encouraged to contact us directly. Participation can be in the form of an appraisal or piloting the use of the model processes, guidelines, training materials, tools, or technology transfer kits we develop. Pilot projects will each be represented on our advisory council. They will help the team set priorities based on feedback from their trial use of Phase II products and services.

Findings and Conclusions

The four pilot appraisals proved beyond a doubt that it is feasible to use the modified SA-CMM (based on our recommended changes) to stimulate increased use of product line, architecture, and software reuse concepts. These appraisals also helped us understand where acquisition organizations need help, especially when they adopt the SA-CMM. These needs translate into the following process requirements for our Phase II efforts.

- Acquisition processes should be compatible with those employed by suppliers, partners, or contractors. Synchronization points between processes used by these multiple parties, such as reviews, should be identified and well bounded.
- For government organizations especially, developed processes should be geared to supporting major program reviews and any funding cycle requirements.
- Leverage over suppliers should be gained primarily via strategic partnerships. Both acquirers and suppliers should invest their own resources as part of such partnerships. For the partnership to work, both sides should gain some advantage from the other. For example, the acquisition office might offer to market a supplier's product internationally if they make such reciprocal investments.
- Leverage over contractors and subcontractors should be gained primarily via controlling the fee allocations and action items from program reviews. Using incentive or award fees to stimulate achievement of a goal should be pursued as part of the acquisition strategy, especially for reuse.
- Because the management of relationships is the key to supplier management, techniques that improve such practices should be highlighted by our work. The strength of the relationship and the degree of confidence and trust that exists between parties effect leverage.

During Phase I, we also found that there were some basic things program offices could do to improve their acquisition management processes. For example, we found that the following guidelines need to be followed as processes are developed to mechanize the SA-CMM.

- Make software reuse a concern in your and your supplier's software development plans.
- Establish a software reuse working group to recommend how to put product lines, architectures, and software reuse concepts to work on the program.
- Incorporate product line, architecture, and software reuse concepts into the checklists you use at reviews (both program and peer).
- Empower your chief engineer to make decisions relative to the refinements and use of your product line architecture.
- Use earned value, technical performance measures, and rate of progress information to determine how well suppliers or contractors are doing relative to plans.
- Strengthen and use risk management concepts to identify, prioritize, and address the top 10 risk items on the project. Factor risk resolution into your fee plans.
- Take advantage of the multitude of public resources that are available within the government and on the World Wide Web for help implementing these ideas (the Army Reuse Center, the SEI, etc.).
- When appropriate, partner with contractors to accelerate their being awarded a higher SW-CMM level. For example, make their training costs allowable under the contract only if they achieve this higher level of process maturity.
- Provide appropriate contractual incentives to stimulate increased levels of reuse.

In summary, our Phase I SBIR effort proved the feasibility of stimulating increased reuse by adding product line, architecture, and software reuse concepts to the SA-CMM. The Phase I effort also demonstrated that there is a market for related goods and services. The pilot appraisals conducted during Phase I helped us develop needs, priorities, and requirements for Phase II. We are encouraged by the results and are trying to

fulfill these needs as part of our current Phase II SBIR efforts. ◆

About the Authors

Tara Ragan is a computer engineer with the Advanced Technology Directorate of the U.S. Army Space and Missile Defense Command. She has 14 years progressive software engineering experience in government and industry, including involvement in all phases of the software acquisition and development lifecycle. She has performed SW-CMM-based Software Capability Evaluations for the Army and BMDO since 1991. She is currently the governmental technical representative on several SBIR efforts, including the subject BMDO-sponsored Phase II SBIR effort with RCI. She has a bachelor's degree in computer science from the University of Alabama and has done postgraduate work at the University of Alabama in Huntsville. She is an Army Acquisition Corps Level III professional, a member of the Institute of Electrical and Electronics Engineers (IEEE) Computer Society, and the Huntsville, Ala. area Software Process Improvement Network.

U.S. Army Space and Missile Defense Command
ATTN: SMDC-TC-AS/Ragan
P. O. Box 1500
Huntsville, AL 35807-3801
Voice: 205-955-3515
E-mail: ragant@smdc.army.mil

Donald J. Reifer is one of the leading figures in the fields of software engineering and management, with over 30 years of progressive experience in government and industry. From 1993 to 1995, he was chief of the Ada Joint Program Office, technical adviser to the Center for Software, and director of the DoD Software Reuse Initiative under an Intergovernmental Personnel Act assignment with the Defense Information Systems Agency. He is president of RCI, which specializes in helping clients improve the way they do business. He has a bachelor's degree in electrical engineering from New Jersey Institute of Technology, a master's degree in operations research from the University of Southern California, and a certificate in business management (MBA equivalent) from University of California at Los Angeles. His many honors include the Secretary of Defense's Medal for Outstanding Public Service, the NASA Distinguished Service Medal, the Frieman Award, and the Hughes Aircraft Fellowship.

Reifer has over 100 publications, including the popular *IEEE Software Management Tutorial (5th ed.)* and a new Wiley & Sons book entitled *Practical Software Reuse*.

Reifer Consultants, Inc.
P.O. Box 4046,
Torrance, CA 90505
Voice and fax: 310-530-4493
E-mail: d.reifer@ieee.org

References

To view the recommendations and final report, visit Don Reifer's Web site at http://www.reifer.com.

1. Paulk, Mark C., Charles V. Weber, Bill Curtis, and Mary Beth Chrissis, *The Capability Maturity Model*, Addison-Wesley, Reading, Mass., 1995.
2. Ferguson, Jack, et al., *Software Acquisition Capability Maturity Model, Version 1.01*, CMU/SEI-96-TR-020, Software Engineering Institute, Pittsburgh, December 1996.
3. Reifer, Donald J., "Recommended Changes to the SA-CMM," *Proceedings of the Reuse '97 Workshop*, Morgantown, W. Va., July 1997.
4. Reifer, Donald J., *SA-CMM Market Survey Results*, Reifer Consultants, Inc., Report RCI-TN-150, June 30, 1997.
5. "DoD Software Reuse Initiative," *Strategic Plan*, June 16, 1995.
6. Reifer, Donald J., *Questionnaire for Use with the Software Acquisition Capability Maturity Model (SA-CMM) "Quick-Look" Appraisals*, RCI-TR-050, Reifer Consultants, Inc., June 30, 1997.
7. Reifer, Donald J., *Final Report: Adding Product Lines, Architectures, and Software Reuse to the Software Acquisition Capability Maturity Model (SA-CMM)*, SBIR Phase I Report (CDRL A002), Dec. 27, 1997.
8. Reifer, Donald J., *Business Plan/Case for Pursuit of Software Acquisition Capability Maturity Model (SA-CMM) Products and Services*, Reifer Consultants, Inc., Aug. 1, 1997.

Using Inspection Data to Forecast Test Defects

John T. Harding
Software Technology Transition

Some organizations have applied software inspections well but may not be using the data to improve the inspection process and to make business trade-offs based on the inspection data. This article describes how to use inspection data from code inspections to forecast the number of defects that remain in the product and how to forecast the number of defects that need to be removed during each test activity.

What Is the Difference Between an Inspection and a Peer Review?

Within the programming community, the term *peer review* can refer to one of many different types of reviews. Some are consistent, rigorous, and have associated data, but more often they are informal, their effectiveness is highly dependent on the people involved, and they rarely have associated data. To eliminate misunderstandings, this article will use the term *inspection* to refer to consistent and rigorous "peer reviews" that have associated data.

What Are Software Inspections?

An inspection is a review of a work product, led by a moderator who is not the producer, that seeks and records defects in that work product using standardized checklists and techniques. The inspection process initiates rework as necessary, initiates re-review, passes the work product based on exit criteria, and adds to the base of historical data.

What Is Needed to Get Good Data for Analysis?

As with any measurement, the data must be consistent and accurate or you will be comparing apples and oranges; however, this does not mean the data has to be precise. Typically, organizations at lower levels of process maturity cannot be as precise as organizations at higher maturity levels. Do not let this stop you from using the data. It is more important to understand the relationships and be in the right order of magnitude when you start to use the data than to focus on getting to the nth digit to the right of the decimal point. Merely focus on consistent data, which requires a consistent process, and accurate data, which requires understanding some common definitions.

Consistent processes require a definition of both the artifacts to be inspected and of the process to create them. This consistent process ensures that the inspected work products are similar. Most organizations evolve to rather than start at this point. Criteria for work products, including coding standards, design standards, guidelines, and templates, can help ensure that the various artifacts are similar, regardless of which person in the organization worked on the artifact.

The need for different people to review each other's artifacts during inspections helps ensure that these criteria and enablers for consistency evolve, especially with respect to design and coding standards that improve consistency and readability. This also requires that the inspection process be consistent, which may be the easiest aspect for many organizations since it is a "single" process. Some of the key aspects are training, project-specific checklists, criteria for reinspection, data definitions, scenarios, establishing project-specific inspection rates, and data-capture mechanisms.

Accurate data requires that some primary data elements be defined and commonly understood, especially the definitions of major defects, size, and time. Even after these elements are defined, people will still need to make value judgments to record data. These judgements are not likely to produce consistent, useful data until there is a common understanding of how the data will be analyzed and used. Most organizations have difficulty getting consensus on what a major defect is until people realize what trade-offs will be made based on the reported data.

If I Have a Consistent Inspection Process and Good Data, How Can I Start to Forecast the Number of Defects Test Needs to Find?

The following example illustrates how this could be done. If you have done some code inspections, the data might look as follows:

- Product contains 10,000 lines of code (LOC).
- Two thousand LOC were inspected.
- Fifty major defects were found in the inspections.

After examining the inspection rates, you determined that they were all reasonable and that the inspections appeared to be consistent. Although there is not one set of numbers applicable to every project, some relationships appear to be consistent across many different software organizations. For example, groups that are starting to do consistent inspections typically discover about 50 percent of defects present in the product. Rarely do they start out finding as many as 60 percent or as few as 40 percent. With this industry norm and the project-specific data, you can project the following:

- Fifty defects in 2,000 LOC is a defect density of 25 defects per thousand LOC (KLOC).
- Assuming 50 percent inspection effectiveness, the product defect density is 50 defects per KLOC.

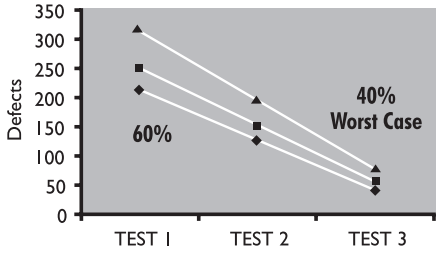

Figure 1. *Estimating remaining defects.*

- The product contains 50 defects per KLOC x 10 KLOC, or 500 defects.
- Fifty defects were removed during the inspections, so there are 450 defects remaining.

You now have a quantifiable target for the number of defects that should be found in test. There are, however, some additional calculations to perform. Since inspections on most projects are initially between 40 percent and 60 percent effective, you can redo the calculations for the 50 percent effectiveness for both of these limits to show a range of defects for each test stage, then expect the actual number to be somewhere in between (see Figure 1).

For the 40 percent calculation:

- Inspection defect density is 25 major defects per KLOC.
- Product defect density is 62.5 defects per KLOC (25/.4).
- There are 570 major defects remaining (10 KLOC x 62 defects per KLOC = 620 defects minus 50).

For the 60 percent calculation:

- Inspection defect density is 25 major defects per KLOC.
- Product defect density is 41.6 defects per KLOC (25/.6).
- There are 370 major defects remaining (10 KLOC x 42 defects per KLOC = 420 defects minus 50).

Now, when the test defect data comes in, you will better understand what questions need to be asked.

Since it is nearly always more cost-effective to find defects in the early testing activities, you probably want to find the largest number of defects in the first test stage and less in each subsequent testing stage. There are any number of techniques to do this, including various deprecation algorithms. You could also just do it manually, such as was done for Figure 1, just by drawing a straight line. As you are doing this, it is OK to round up or down—this is "blackboard" mathematics.

If there are roughly 450 defects remaining, you can decide how to remove them by comparing the time and cost to remove them using inspections with the time and cost to remove them using testing. Typically, inspections will be much more cost-effective. Therefore, you might decide to perform more inspections for the other 80 percent of the product that has not yet been inspected to reduce the number of defects going into test. If you choose to go directly to test, you could estimate how many defects you need to find during each test activity (see Figure 1).

Inspection Data from Additions and Modifications

If you are adding or modifying functions in an existing product, you will need to gather some additional data. When you inspect changes, you will typically have to examine some of the unchanged code to understand how the changed code works. This analysis will require that you count both the total LOC inspected and the new or modified LOC inspected. The major defects will also need to be collected at the same level of granularity, which helps you understand both the LOC and the defects associated with the added or modified functions. The example in this article assumes that the LOC and major defects are both associated exclusively with the count of new and changed lines of code.

Defect Density Considerations

There is one additional calculation you should consider. What if the code that was inspected is from the least defect-dense parts of the product? What if it is from the most defect-dense parts of the product? How many of your products have a constant defect density across all parts of the product? You should try to understand if the inspection data represents a sample of the product with the same distribution of defect density as the entire product. If you have complete data regarding customer, inspection, and test defects, this can be calculated easily. If you do not have this data you may need to do some additional "sampling" inspections to determine which parts of the product are high and which are low with respect to defect density.

If there is no defect density information available, a simple approach is to divide the product into low and high defect-dense areas based on asking the programmers in which category each file or module belongs. These "sampling" inspections also provide you with the defect density values. Then you could redo each of the calculations (40 percent, 50 percent, 60 percent), and instead of assuming a constant defect density, you could apportion the product into different defect densities and calculate the number of defects in each different portion of the product. Table 1 illustrates how this could be done at 50 percent effectiveness. The total defects remaining is 410 (+ 280 from low density + 180 from high density - 50 found during inspections). Please note that if you have a significant amount of defect density data, you can break your product into 10 decile ranges for this calculation.

The Bottom Line

When you first start to do this type of analysis, you may feel a little uncomfortable using inexact numbers. You need to ask yourself whether you are better off

Table 1. *Defects apportioned by low and high density, assuming 50 percent effectiveness.*

Defect Density Group	Lines of Code	Defects per KLOC (from inspections)	Defects per KLOC (for group)	Total Defects
Low	7,000	20	40	280 (7 KLOC x 40 defects per KLOC)
High	3,000	30	60	180 (3 KLOC x 60 defects per KLOC)
Total	**10,000**	**50**	**100**	**460**

trying to show business value with these rough numbers. I believe most executives would prefer to use inexact data and try to refine it over time. Software inspections are valuable, but they take time; it is important to maximize the business value your organization can achieve from them.

How Do I Optimize the Time Spent on Inspections?

For most software development projects, the single largest cost factor is labor. If doing inspections takes time, optimizing the time spent on inspections means getting the most impact from each inspection by finding the most defects per unit of time expended. Although this may appear obvious, many organizations are not doing this.

Another aspect is to focus the inspection process on finding the most significant product defects. "Significant" would have to be defined by members of the project to include cost to fix, critical functions, or other similar attributes. Analysis of the defect data from inspections, test, and customer-discovered defects can help effectively focus the inspection effort. This is done by understanding and analyzing the types and characteristics of these defects and then modifying the inspection process, e.g., updating checklists, additions, or modifications to scenarios. Inspection data can also help determine the optimum planning rates for the inspections, provide insights into areas for process improvement, and help to build defect removal models.

Additional Elements for Successful Inspections

Champions

As people try to implement any new technology, they need a focal point in the organization who can help them tailor the new technology to meet their project-specific needs. This person also ensures that commonly occurring problems are solved in a consistent manner. The champion may also have to keep the organization focused on the new technology to keep it "alive" until it is a permanent part of the infrastructure.

Moderators

Skilled moderators are key to ensuring consistent control and focus across different inspections. They must be able to manage their peers and keep the discussions focused on finding defects.

Metrics Policy

Because the goal of inspections is to find defects, then to analyze the defect data, there is a need to "decriminalize" defects. A metrics policy statement may help accomplish this.

Time

Given the time required to perform truly effective inspections, the time and resources must be planned, or the inspections are unlikely to happen. You must also be selective in what you inspect so that you focus your efforts on the areas with the highest potential payback.

Data Analysis and Feedback

If people think data are being collected but not used, you may get "garbage in" data. People will do a better job capturing and recording data if they know how the data will be used and how it will impact them. Data recording often requires some degree of interpretation, and the associated value judgements will be much more consistent and accurate if the people doing the data gathering are closely involved with the analysis and its resulting decisions.

Some Basic Analysis Concepts

Although this article is not long enough to fully elaborate on the analysis concepts, a few aspects should be highlighted:

Consistent Inspections and Consistent Data

This is critical if the data are to be used to help make decisions. Consistent processes and data are key to having consistent inspections.

Questions

The key to data analysis is to help people on the project understand what questions they need to ask. In many cases, the data will not tell you what to do, but rather where you need to probe for more information.

Accuracy vs. Precision

Many people with technical backgrounds are uncomfortable using imprecise data. Accuracy means the data truly represent what you believe they represent. The focus on accuracy and not on precision should be publicly discussed so that everyone understands the objective.

Timely Feedback

Analysis and decisions need to be made during the project—do not wait until the end of the project, when it is too late to make decisions that can impact the project. When an inspection effort is first undertaken, the results should be quantified and presented publicly while the project is still in progress.

Feedback and "Feedforward"

Most people are familiar with feedback, which is typically how lessons learned are captured for application to *future* projects. Feedforward is the use of data from an *early* process activity to adjust how the work is done in subsequent process activities on the *same* project. For example, if design inspections discover that a new or modified function is more defect dense than the other functions in that release, more emphasis can be placed on the defect-dense function during code inspections and test.

Ratios

This is similar to the way financial analysts look at business data. They do not look at any single number to determine a company's financial health, but instead look at a large set of numbers. More important, they examine the ratios. For inspections, the following key ratios should be examined.

- Units of size per hour of inspection. This could be pages, LOC, or diagrams per hour, or whatever is appropriate for the situation. These rates should be examined to determine if they are reasonable for the type of material being inspected. This data can also be used in subsequent analysis, but only if it is accurate and reasonable. I have seen organizational

data that suggest inspections were held at rates much too fast for optimum efficiency, sometimes approaching 1,000 LOC per hour. In some cases, the inspectors are just recording the total size of the material, not the amount of material inspected. Rates need to make sense, which requires consistent ways to count both the time and the size. You can use standard reading rates to determine if the rates observed are reasonable.

- Defect density. This could be major defects per page, per KLOC, or whatever is appropriate. For organizations just starting to do code inspections, defects per KLOC is usually appropriate. As organizations reach improved quality levels, some are starting to measure defect density per million LOC. If LOC is the measure, the data should exclude comments or anything that would not change the compiler-generated object code. Defect density can be considered the "yield" from inspections. Since finding defects is the goal, finding more of them should be an objective. Defect density information can help organizations determine where to inspect.
- Hours per major defect. Since labor is the largest cost factor for most software projects, measure the hours it takes to find and fix problems—this figure will be more meaningful to the workers and managers. If you know the hours, you can apply labor rates to arrive at dollars. The ratio would be the sum of all hours expended for the inspection divided by major defects. Typically, groups starting out will find major defects at a rate between two and four hours per major defect. That rate may go much lower, especially for code inspections.
- Major defects to minor defects. This ratio can help you better understand if the focus of the inspections is on major or minor problems. It may also highlight recording problems.

Basic Analysis Techniques to Optimize the Time Spent During Inspections

Of all the techniques available, scatter charts are probably the easiest to use and may provide the most useful information. The key is to keep things as simple as possible and not look for complex relationships. Because inspection data represent an intellectual activity and not a mechanical operation, the data will have a higher degree of variance than typical factory data.

Avoid overly sophisticated tools. Many textbooks suggest that linear regression techniques require high correlation coefficients if a relationship exists between the two variables; however, you may find that for inspections and other software-related data, the coefficients may be lower than what the statistical books recommend. Does this mean you should not use the data and the relationships? No, it means you need to be careful. The common relationships that appear in most projects are discussed below. Consider starting to use scatter charts at a project level where the data represent a somewhat consistent entity. Consider starting to analyze the following:

- Major defects per KLOC vs. LOC per hour (do one for inspection time and another for preparation time).
- Hours per major defect vs. LOC per hour (for inspection).
- Size of material inspected vs. hours per defect.
- Size of material inspected vs. defects per KLOC.

Defect Removal Models

Once you have optimized the inspection process, consider developing a defect removal model to help with planning, tracking, and identifying areas for software process improvement. A defect removal model will allow you to understand the process capability for defect removal during each process stage.

Won't Analyzing the Data Only Take More Time?

You could just fix the defects found during inspections and not record data, but there are many important uses for inspection data, many of which are explained throughout this article. Not the least of these uses is to maintain management support for inspections. Look at how much time you may be investing in the inspections alone:

- The typical rate for code inspection is 125 LOC per hour.
- The typical number of inspectors is four.
- Preparation time ideally equals inspection time.

Although these values may make inspection appear more time-consuming than expected, data from many different types of organizations show these values fall into acceptable ranges. The key is for each project team to identify the optimum rates for its project. This means that if four people inspect 1,000 LOC at the above rate (4 x (1,000/125)), it will take 32 hours for inspection and another 32 for preparation, which is 64 hours or approximately 1.5 people weeks. This is not a trivial investment, especially if a significant percentage of the artifacts for a software development project are to be inspected.

When you look at these numbers, you may be ready to cancel your inspection program, but wait—the cost of removing defects through inspections needs to be compared with the cost of removing the same defects in test or having to fix them when they are found by customers. Many articles have shown that organizations that analyze their data can find and fix defects much more cheaply with inspections than during subsequent process activities. Many organizations have shown that they can find and fix the average defect in two to four person hours or less using inspections, which is typically much faster than most testing processes. Although your numbers may not be the same, you will not know unless you analyze your data.

It should only take a couple of days to do this analysis, which is well worth the time for the improvements that can be effected based on this analysis—especially given the amount of time spent on inspections. And the bottom line is managers and executives are unlikely to continue spending money on

see HARDING, page 24

Year 2000 Automated Testing
A Summary

Don Estes
2000 Technologies Corporation

For many organizations, it is neither practically nor financially feasible to test all year 2000 program fixes to a level of accuracy that approaches 100 percent. However, there are guidelines to determine whether a system, when it is fielded, is likely to have a containable level of faults. Several automated testing techniques can help attain this level of assurance while helping work within time and financial constraints. This is a summary of a much longer working paper that can be found in the May 1998 Internet version of CROSSTALK at http://www.stsc.hill.af.mil/CrossTalk/crostalk.html.

Testing applications for year 2000 (Y2K) compliance is the equivalent of seeking a safe harbor for an anticipated storm. The metaphorical storm will be the organizational and economic disruption that could result from the failure of critical computer systems to function correctly as a result of the date change from 1999 to 2000. The safe harbor thus sought is to ensure, in advance, that the computer systems within the control of the organization will function correctly and will therefore ensure the business against loss.

The complete version of this article details the steps that must be taken to enter that safe harbor or, if this is impractical or inadvisibly expensive to reach, to arrive as close as possible with available resources. However, the scope of the complete article is limited to testing internal application programs. Application systems sited within the organization, if found to be at fault, can be repaired either with in-house or contractor staff or with staff of the licensing vendor. Application systems sited outside the organization that interface to internal application systems must be examined to ensure the data they send and receive remain compatible and that appropriate actions be taken with the correspondent party if incompatibilities are found and not fixed. Nonapplication systems, if found to be at fault, can in most cases only be replaced.

The primary issue of application testing is that it is not mathematically feasible to test all programs to a level of 100 percent certainty. It may be possible to get close enough to 100 percent to constitute no practical difference—or the scale of the task, the elapsed time required, or associated costs may be so great that significantly less than 100 percent may be the best that can be realistically achieved. Thus, there is the preferred case of risk minimization, which is close to 100 percent, vs. the alternative case of risk optimization, which tries to minimize the risk to the business while accepting that some level of risk will be unavoidable.

Ultimately, the problem can be reduced to a risk vs. cost trade-off. Widely quoted statistics put the cost of testing for a Y2K project at 40 percent to 60 percent of the total cost of the project or roughly a 2-to-1 or 3-to-1 ratio over the cost of renovation. Low accuracy requirement applications, such as noncritical government service applications, may require less than a 1-to-1 ratio of cost of testing to cost of renovation. Conversely, where extremely high accuracy is required, as is frequently the case in the financial industry, the cost can exceed 80 percent of the total cost or more than a 5-to-1 ratio of the cost of renovation. The higher the accuracy required and achieved, the lower the resulting risk of business disruption but the higher the cost. It has been frequently observed that Y2K testing projects lack sufficient resources for testing to even a modest level of accuracy, and reaching the business case level of accuracy may require substantial increases in resources.

Testing is required to ensure that Y2K compliance modifications made to programs do not introduce new problems and to assure that the programs will continue to operate correctly as the data they process begins to include dates in the 2000s as well as in the 1900s. However, what will be the consequence of inadequate testing? For mainframe and client-server systems, undetected residual program faults will show up in one of two ways:

- Outright program failure, forcing a halt to at least some part of the application processing until the program can be repaired.
- Data corruption, which will force the application completely off-line until the program problem causing the corruption is traced, repaired, and tested and the data errors are repaired.

It is important to keep in mind that any significant data processing installation has some level of faults, as any user of desktop software is reminded daily. However, most faults are too trivial to worry about. Provided there is sufficient staff to cope with nontrivial problems and there is contingency backup for rare crisis situations, it is a *containable* level of faults.

The difference in a Y2K application failure situation is a matter of degree, not of kind. The level of daily faults will reach a point that will overwhelm the support staff; the contingency backup support, which is designed for isolated crises, will also be overwhelmed by simultaneous crisis calls from too many sites. The faults will come in waves as critical dates are reached for each application, and the faults will build to a peak around the end of 1999 and the beginning of 2000. Faults will start to recede after March 1, 2000, although new failures will continue for some time. We are already seeing a few cases of significant Y2K faults, although so far none have been overwhelming.

If a testing project fails to complete full testing, it does not *necessarily* follow that the renovated application will fail in production. In some cases, the level of undetected faults will be containable in practice. In other cases, undetected faults will not be containable, and damage to the business will result. The business purpose behind significant testing projects is to take chance out of the equation and to provide an insurance policy against damage. In this sense, the cost of the testing project can be considered the premium on an insurance policy.

The complete version of this article details what is required to achieve risk minimization using conventional testing methods, how to proceed in a risk and cost optimization testing project using conventional testing, and a discussion of some innovative technical approaches to introduce economies of scale by automating the process of testing. Where applicable, automated testing can allow a testing project to move significantly closer to the risk minimization model within the limits of what is practical and affordable. ◆

About the Author

Don Estes is chief technology officer for 2000 Technologies Corporation, for whom he has designed and implemented both a data encapsulation and an automated testing system. He also works closely with vendors of limited window-

ing, program encapsulation, and object code remediation systems. He has been involved with COBOL and database applications for 25 years and database and mainframe performance tuning for 10 years. For the last seven years, he has helped design and execute projects for the mass modification of large bodies of source code, primarily for platform migration, using state-of-the-art automated source language transformation technologies and automated testing methods. He is a regular contributor to Peter de Jager's Year 2000 mail list, where he is known for his contributions relating to Y2K rapid compliance strategies and automated testing. Estes is a graduate of Massachusetts Institute of Technology in physics, with a postgraduate degree from the University of Texas in educational psychology.

2000 Technologies Corporation
114 Waltham Street, Suite 19
Lexington, MA 02173
Voice: 781-860-5277, 800-756-8046
E-mail: info@2000technologies.com

HARDING, from page 22

inspections unless they start seeing value for the dollars spent. They need to see the business payback in quantitative terms. The business value analysis should compare the cost (hours per major defect) for defects found in inspections with the cost for each test activity. If you have not been collecting this data for the test activities, you may need to have developers and testers estimate the number of hours they believe it takes to find defects during each test activity. ◆

About the Author

John T. Harding is one of the founding partners of Software Technology Transition, which provides training and implementation in the Software Engineering Institute (SEI) Capability Maturity Model (CMM) and CMM-Based Appraisal for Internal Process Improvement method and in software inspections, metrics, and project management. Other work includes the International Organization for Standardization (ISO) gap analysis, helping organizations develop business and software baselines, and action planning for software process improvement. He was a visiting scientist at the SEI, was the metrics mission manager for Groupe Bull, and held various technical and managerial positions in software development with IBM and the Bank of Boston. He has a master's degree in business administration from Boston University and a bachelor's degree from Rensselaer Polytechnic Institute (RPI) and is a member of the Association for Computing Machinery and the Institute of Electrical and Electronics Engineers.

Software Technology Transition
60 Elm Street
Andover, MA 01810
Voice and fax: 978-475-5432
E-mail: johntharding@compuserve.com

Recommended Reading

1. Ebenau, R.G. and S.H. Strauss, *Software Inspection Process*, McGraw-Hill, New York, 1994.
2. Gilb, T. and Dorothy Graham, *Software Inspections*, Addison-Wesley, Reading, Mass., 1993.
3. Weller, E.F., "Lessons From Three Years of Inspection Data," *IEEE Software*, September 1993, pp. 38-45.
4. Weller, E.F., "Using Metrics to Manage Software Projects," *IEEE Computer*, September 1994, pp. 27-33.
5. Grady, Robert B. and Deborah L. Caswell, *Software Metrics: Establishing a Company-Wide Program*, Prentice-Hall, Englewood Cliffs, N.J., 1987.
6. Grady, Robert B., *Practical Software Metrics for Project Management and Process Improvement*, Prentice-Hall, Englewood Cliffs, N.J., 1992.
7. Ishikawa, Kaoru, *Guide to Quality Control*, Asian Productivity Organization, Tokyo, 1976.
8. Burr, Adrian and Mal Owen, *Statistical Methods for Software Quality*, International Thomson Computer Press, London, 1996.

AFMC Production Acceptance Certification and Depot Maintenance Quality Software Project

K. Edward Lynch
Advanced Programming Institute, Inc.

Amid the thunder, flash, and crash of the monster, multimillion- and billion-dollar software projects wherever we look, an occasional small, productive step forward is taken. This article is about a success story on many levels. Although this project was relatively small by most software project standards, this article discusses the successes surrounding the software system, developed in-house by an Air Force Materiel Command (AFMC) organization, supporting a 24-hour-a-day, 365 days-a-year, real-life, in-the-trenches mission in the AFMC depot maintenance arena.

Among the many challenges faced by first-level depot maintenance supervisors in an Air Logistics Center (ALC) is to maintain the individual records (task and skill certification and completion of training) for the mechanics or technicians under their supervision in accordance with AFMC Instruction (AFMCI) 21-108, Production Acceptance Certification (PAC). This is a daunting task given the thousands of tasks associated with performing the overhaul and maintenance of U.S. Air Force and foreign air forces' equipment of every type. It is even a more daunting and crucial task in this era of rapid changes and diminishing resources. Mechanics' certification of skill to perform a task and the sign-off on their work have far-reaching and significant consequences. Many lives may be held in the balance by equipment that is installed or repaired by these workers.

Why Another Software System?

In 1993 there began an effort to produce a standard, command-wide software system to manage the data concerning the tasks and training of skilled mechanics and technicians. At this point, it had become clear that software systems developed at several ALCs used older software technologies limited in their ability to deal with changes on the horizon. Newer software and hardware technology began to impact operations at every level as operational workload was redistributed, and the availability of technical skills was subject to a flux not experienced in decades.

Factors that Contributed to the Project's Success

The success of this software project is due to the participation of many users who contributed throughout the process. Although the project did not formally adopt a rigorous software lifecycle development method, in practical ways, the key elements for developing any good software were used during the process:

- Develop the requirements as quickly and clearly as possible.
- Use good tools that can keep up with the changes in technology.
- Involve the users and testers all the way through the process.
- Review! Review! Review!

More than anything else, it is clear that people determine the success of the software project—it is the combined effort of user and developer. Good software technology is important in any software development endeavor, but success is ultimately the product of effective communication and interaction among the people who do the functional and technical work. The lure of technology and the dizzying pace at which it changes dazzles most of us in the development trenches; often, keeping up becomes an end in itself. But great technology and brilliant developers can produce fabulous software that is never used because it does not conform to the real needs of the user.

However, the user can tolerate less than the most current technology and less than breathtaking design if the new software delivers the means to do more work better, faster, and cheaper. When the software forces the user to rearrange a work practice or takes a long time to learn, it will be resisted or, even worse, sabotaged, especially if the user feels that requirements based on the business objectives were not properly addressed in the software development. When the software does deliver those requirements, the user welcomes the better tool as long as the payback from converting to the new tool is greater than the effort.

Involve the User in Requirements Development

The first of many successes of this development project was produced by the initial project manager and development team leader. It is tough enough to get requirements from users who are all in the same vicinity, but getting requirements from users in widely separated geographical areas who have vastly different workloads and practices was one of many challenges the development team overcame. They went to the users to find out what they wanted the software to do, and the team went back to the users for periodic reviews over the life of the project.

As a participant in many of the later reviews, I was delighted that users and coordinators from the ALCs came to provide real participation and contributions to the end product. At no time were any of these users bashful about

expressing their needs. As a result of these reviews, many functional capabilities were incorporated to facilitate the movement of data between supervisors and to handle the association of training requirements with specific tasks. Additional capabilities were developed to allow supervisors to set up "templates" of training and tasks that are used to quickly incorporate new employee data in the system. In many areas, this system supplements training tracking and training management capabilities found in other systems: Some users anticipate using this system for training management when support for other systems is terminated.

Involve the User in Frequent, Regular Reviews of the Project Process

A series of reviews, held at six- to eight-month intervals at the ALCs and at Headquarters AFMC, Wright-Patterson Air Force Base, were accomplished using the demonstration of a progressively developed prototype to stimulate the evaluation and let participants see for themselves what progress had been made on the project.

Plan to Revise the Requirements During the Early Stages

The initial requirements specification, no matter how thorough, rarely includes everything. The project's users became smarter in that process and came up with better ideas for doing the work. Sometimes the developer suggested alternatives that users found acceptable. During the review process, all capabilities that did not meet the users' expectations were documented, and the requirements and design documents were revised to better reflect the users' expectations. All remaining items scheduled for future development were reviewed and priorities re-established as requirements and design issues were clarified. These items became the agenda for the next review.

Run-time copies of the software were made available to users who wanted to independently do testing and review. The emphasis was to get the product into the hands of the users as soon as possible to enhance the prospects of receiving good feedback early in the process. The tendency for requirements creep dropped off after the third or fourth review session. This user feedback process continued to function effectively despite three changes of the AFMC headquarters office of primary responsibility, three changes of the primary contractor, one change of the General Services Administration (GSA) administrator, three changes of the ALC project manager, three changes of the development team leader, and several changes of development team programmers. During all these changes, GSA administrators still regularly reviewed project finances and progress with the contractor administrators and the AFMC project managers.

Use Good Development Tools

The second success was the decision to use a new (at the time) software development tool[1] that enabled the developers to produce a progressive Windows prototype in a Windows environment, which also enabled a stand-alone version to become a client-server product with minimal conversion effort. This tool provided many of the object-oriented features that would assure a long and healthy life for this software. Early choices of help authoring,[2] database design,[3] and distribution and install software[4] also proved to have the power to support the project over the long haul and through several changes of software technology and requirements. These tools enabled a small number of developers to accomplish the work in a reasonable length of time.

MIL-STD-498 was another tool that was key to the success of this project. It provided an extensive structure of guidelines for system requirement, design, user reference, and technical reference documentation, which allowed the developers to produce coherent and relevant documentation for the system. Having a wide spectrum of project components from which to choose assures that all relevant project aspects were considered for this project. It provided a ready sanity check to assure that all relevant project components were considered as the project progressed. It also provided assurance to the users that all relevant components were considered, along with a physical document they could evaluate for verification of specific components.

Involve the User in Testing as Early as Possible

The third success was the ongoing and early involvement of key users and trainers at each ALC. They were the field testers who provided the development team with crucial feedback at all stages of the project. User tutorial and guide documents were developed at the same time features were established and tested, enabling quick training on the current build of the software.

Train and Support the User as Early as Possible

The fourth success was an early "train the trainers" program conducted at each ALC to maintain user support until deployment was accomplished in mid-1996. Working through a network of designated key contacts at each ALC,[5] issues were reported to the development team for timely resolution. During the development phase, fixes were sometimes provided within hours. Post-deployment fixes are issued at quarterly intervals. Support personnel at each ALC download the software and fixes from an FTP site established at the development ALC site.

The project was structured into four software module phases: the first was used to develop the base module to manage the Production Acceptance Certification as a stand-alone system; the second, to deploy the base system as a client-server system; the third, to develop an interface to another command training system; the fourth, to develop a module to collect data for process improvement evaluation, also required by AFMCI 21-108.

System Features

The functions and capabilities of the main module support the procedures to maintain AFMC Form 75, Job Knowledge-Training Certification Standard. All capability is designed to be accom-

plished by the first-level supervisor. Additional functionality was added to support a number of administrator capabilities that are especially important in the client-server environment.

- Personnel data.
- Task assignment data.
- Training data.
- Supplemental training data.
- De-certification of tasks.
- Supervisor history.
- Task data, which assigns required courses to specific tasks.
- Course data management.
- Standard reports, transportable among site and supervisors.
- Task-training template.
- Ability to convert data from four pre-existing software systems.
- On-line user guide and help.
- Backup/restore utility: a single-click backup and restore process for stand-alone PC users developed using third-party software[6] that integrated smoothly into the application system.
- Work center administrator: a separate set of utilities that perform across all work centers and convert data from four other programs.
- System security access management features.
- High- and low-level reports.

New and Planned Features

Recently, several new capabilities have been added to enhance the client-server version, and we are in the beginning stages of upgrading and optimizing to 32-bit technology and more current object-oriented coding technology. A recent addition allows supervisors to use a personal identification number to sign off on annual reviews electronically, eliminating volumes of paper files. Talks have also been developing in AFMC for an interface to a major production planning and control system that requires current certification data before work can be assigned to available mechanics. A broader course management module has been added to simultaneously support completion updates across multiple work centers. Many edit functions can now only be accessed by delegated administrators.

The third phase of the project, the interface with a command-wide training management system, is awaiting the completion of the most recent rewrite of the system, from a 1960s software technology to PowerBuilder 5.0.

The fourth phase of this project, a process improvement data collection module, is also a requirement of AFMCI 21-108 and supports the recording of data about depot maintenance processes that enables the user to evaluate process effectiveness and analyze areas where process improvement can be pursued.

Conclusion

As of mid-1997, the software had been deployed to all the depot maintenance ALCs and is widely used in the stand-alone mode. A division at one ALC has deployed the client-server version; other ALC divisions are preparing to follow suit. Over 1,000 supervisors are administering the records of several thousand employees at this time. Supervisors now maintain a large number of records for employees more accurately with a minimum of time and effort, and records can be moved from one supervisor to another with minimum effort.

This deployed software product may not be glamorous, but it

- has broad scope and impact at the working level.
- addresses day-by-day needs of first-level supervisors to facilitate the accomplishment of essential work.
- was developed with plenty of user participation and contribution.
- uses resources sparingly but effectively.
- provides a common, command-wide, standardized tool to maintain this data.

Whatever other legacy this project effort produces, all who have been involved can take pride in these accomplishments alone.

It is amazing that developing the technology to produce the blueprint for a software project has been so long in happening; nevertheless, there have recently been some encouraging technical advancements in the acquisition of system requirements, the establishment of design specifications, and in modeling software systems. This is an exciting time to be in the software industry. Let the revolution begin. ◆

About the Author

K. Edward Lynch is a senior consultant with Advanced Programming Institute. He has over 20 years experience in all phases of software development for financial and business systems in mainframe and PC environments for American Airlines, Lockheed Missiles and Space Co., U.S. Army Logistics, Failure Analysis Group, California state and county organizations, Computer Science Corp., Ogden Government Services Corp., and Anteon Corp. He specializes in improving methods related to the critical earliest phases of software development, including management sponsorship, user participation, risk assessment and mitigation, and testing criteria.

Advanced Programming Institute, Inc.
SM-ALC/FMDD
Building 200, Room 230
3237 Peacekeeper Way
McClellan AFB, CA 95652
Voice: 916-643-6493 ext. 281
DSN 633-6493 ext. 281
Fax: 916-643-2217 DSN 633-2217

Notes

1. PowerBuilder from Powersoft, Inc. with a Watcom 4.0 database for stand-alone deployments, Sybase SQLAnywhere 5.0 for the client-server.
2. RoboHelp by Blue Sky Software.
3. ERWin/ERX for PowerBuilder by Logic Works, Inc.
4. InstallShield by Stirling Software.
5. Many individuals at the U.S. Air Force Materiel Command (Wright-Patterson Air Force Base) and at the ALCs located at Davis-Monthan, Tinker, Hill, Kelly, Warner-Robins, and McClellan Air Force Bases. The services of Computer Science Corp., Ogden Government Services Corp., Anteon Corp., and Advanced Programming Institute were provided through a Requirements Contract with the U.S. General Services Administration, San Francisco, under the guidance of Doris Lynch, and earlier, Paul Gurian. A more extensive list of key participants and current contact information can be found at http://emerald.mcclellan.af.mil:2010
6. DynaZip-16 - Inner Media, Inc.

Quietly Making Noise

A Parrothead's Look at Software Process Improvement

Paul Kimmerly
Defense Finance and Accounting System, Financial Systems Activity, Kansas City

The organizational changes associated with software process improvement involve much more than just the technical aspects. The changes that affect the culture of the organization and the impact of those changes on the people involved are often overlooked. This article discusses some of the author's experiences in facing those people-oriented aspects of software process improvement by relating them to the lyrics of singer-songwriter Jimmy Buffett.

Every good cause should have a poet or songwriter to inspire its followers. Software process improvement (SPI) is no different. I propose the music of Jimmy Buffett, the Florida-based purveyor of laid back, Caribbean-flavored music. As a loyal fan (we call ourselves Parrotheads), I wear floral print shirts on a regular basis and go to his concerts locally and around the country. Much to the chagrin of the other members of my software engineering process group (SEPG), I have brought this off-center viewpoint to the office and applied it to what we do. In the following sections, I will share some of Buffett's lyrics to illustrate some lessons learned in our SPI efforts.

"You've got to roll with the punches, play all your hunches, and make the best of whatever comes your way. Forget that blind ambition and learn to trust your intuition. Plowing straight ahead, come what may."
– *Cowboy in the Jungle*[1]

This is probably the first lesson any SEPG learns. Things often do not go as you planned, but occasionally a little opening will present itself and you need to make the best of it when it does. Blind adherence to the model you are following or to any plan you have set up may limit what you can do. Intuition and hunches play an important role in adapting to the changing environment you find. Often, the organization's culture will surprise you and cause you to take a different approach. The trick is to stay light on your feet and adapt to what situations present themselves but never lose site of the goal you are moving toward, and do not let your efforts slip. Plans and processes should be regularly reviewed against the changing environment to ensure you maintain your straight course.

"I'm growing older, but not up. My metabolic rate is pleasantly stuck."
– *Growing Older But Not Up*[2]

This is the anthem of the organization with which you are dealing . In particular, the middle managers of the organization will present this resistance to you. Remember, they may have gotten to their positions by way of the status quo and are possibly being rewarded for maintaining the status quo. These factors make process improvement an extremely tough sell for them. There are three things that can break this barrier: (1) change the reward system, (2) obtain strong senior management sponsorship, and (3) find out the middle managers' problems and identify how process improvement will help solve them.

The reward structure may not be the SEPG's to address. As in many change-related activities, senior management plays a critical role. The senior sponsor must begin to reward the desired behavior and question the status quo. This is a delicate proposition because the day-to-day work must still get out. During the initial stages of process definition and implementation, productivity can drop off, which will fuel the flame of resistance. One key is to reward process improvement-related behavior in the same way other work efforts are recognized. A big danger lurks in allowing the perception that improvement-related activities are separate from "real" work.

"They're just changing channels, waiting for the sails to fill. They'll be changing channels, always will." – *Changing Channels*[3]

Senior management sponsorship cannot be stressed enough. Sponsorship requires a combination of commitment and involvement. Many sponsors are great at committing; they will commit their organizations, commit their resources, and commit their rhetoric. What they need to commit is their time. It is easy for them to say, "Here's a bunch of resources; tell me when we're better." It is difficult for them to commit their own time to get involved in the improvement efforts. If they do not get involved, they will have a committed boat but no wind to fill the sails.

Our organization's headquarters showed its level of commitment to process improvement, causing our director to follow suit. Once we began meeting with him on a regular basis, he began to understand the issues and to address problems we were facing. An-

other issue common to senior managers is their mastery of sending conflicting messages. In the ideal situation, senior managers would ask about software quality assurance reviews with the same conviction they ask about a project delay. Most often, that will not be the case, and process improvement issues can take a back seat to the crisis of the moment. The SEPG must work with its senior manager to point out the conflicting messages and minimize the situations where they can be issued. Although the SEPG cannot make the wind of change blow, it can certainly create favorable conditions.

"Something like a Swiss Army Knife. That's my life."
– *Schoolboy Heart*[4]

When dealing with middle managers, the SEPG cannot bring too many tools. Like the above mentioned knife, the SEPG must be prepared to fit any situation that presents itself. Middle managers bring a variety of needs to the table. They relate closely enough to the practitioners to be concerned with technical issues, yet they must respond to senior managers on a higher level. They can be a source of great information and of great frustration. Spending time talking to them and, more important, listening to them can be invaluable. The SEPG must identify the clues that will lead to middle manager buy-in. Once identified, the SEPG should target the concerns that, if remedied, can bring about the biggest pay-off. By concentrating on those, the SEPG can begin to get the middle managers involved. If senior management proves they are willing to reward middle managers for participating in improvement efforts, momentum can build quickly.

You are probably thinking, "That's all well and good, but what about the one person in the corner who keeps saying nothing will work?" Rest assured, such people exist in every organization. Some will never buy in to what is going on, but we have found some success by turning things around and making these people "devil's advocates." Establish that role upfront and tell them the rest of the group will determine a solution and send it to them to identify the problems with it. This channels their negative energy toward helping the group and gives them a specific role to fill. We have found that this gradually leads to more involvement, except in the most extreme cases.

"Who's the blond stranger that entered my life? Making me over in the moonlight."
– *Who's the Blond Stranger*[5]

This is often asked when the SEPG starts its improvement efforts. The SEPG should make itself visible to the organization by giving its members some initial orientation on what process improvement is and what it will involve. Senior managers play an important role here as well-they should identify a specific set of criteria when appointing SEPG members, including good people skills, a desire to effect change, and respectability within the organization. When introducing the SEPG to the organization, senior management should make the selection criteria known, which will help the group's credibility. The process improvement efforts should be treated like a project by management and given the same type of visibility as the development projects. If people are still asking who is in the SEPG several months into the improvement efforts, some catch-up work is needed to make things more visible.

"Quietly making noise, it starts with kindergarten toys."
– *Quietly Making Noise*[6]

Often, the process improvement efforts start quietly by organizing people into work groups to address specific issues. As the group comes together, people will bring biases from their work backgrounds and different levels of acceptance to the efforts at hand. One thing we found to be effective in meetings is to bring toys. The toys provide an icebreaker as people enter the room. They help create a casual atmosphere and often start the group interaction before more serious issues are brought up. This can lead to a more open discussion. Most people prefer items that can be thrown at each other, like soft-textured balls. Balls, spring toys, and other small toys help loosen up the mood. There is just something special about watching two senior managers fight over a child's toy.

"It's my job to be cleaning up this mess and that's enough reason to go for me. It's my job to be better than the rest and that's a tough break for me."
– *It's My Job*[7]

The SEPG holds the unenviable position of sitting somewhere between management and the practitioners. As a result, it must address the issues faced by each group. A lot of the problems we found in our organization were caused by the interaction between management and practitioners and the distance that existed between the two. One key was for the SEPG to operate at a higher process maturity level than the rest of the organization. We had to apply the Capability Maturity Model disciplines to our own activities and develop processes for the activities we perform. To be taken seriously, we had to exhibit the behavior we expected them to adopt.

"Changes in Latitudes, Changes in Attitudes. Nothing remains quite the same. With all of our running and all of our cunning, if we couldn't laugh we'd all go insane."
– *Changes in Latitudes, Changes in Attitudes*[8]

As time passes and various "latitudes" are crossed along the course of process improvement, a number of things change. Attitudes are foremost among them. People begin to see the advantages of doing business a new way and efforts move from the early adopters to the majority of people affected by the change. As latitudes change, new and different challenges present themselves. The SEPG must adapt to the changing situation and draw on their cunning and imagination to face the new challenges.

Along this course, the SEPG needs a sense of humor-it should not take itself too seriously. SPI is a serious matter, but the approach needs to have a light touch. There will be false starts and roadblocks that must be dealt with along the way, and the SEPG cannot let those obstacles bring them down. It is rare that things go exactly as planned, but the SEPG must accept any progress that is made. We learned that SPI is a series of little victories for the SEPG that can lead to big victories for the organization.

> "The years grow shorter not longer, the more you've been on the road. Feelings for moving grow stronger, and you wonder why you ever go home."
> – *Wonder Why You Ever Go Home*[9]

The above represents the desired attitude of the organization after process improvement efforts have taken hold. The time between improvements shrinks as people see the benefits of previous changes. Over the course of time, people begin to look for ways to move to new, improved methods of doing business, and the old ways begin to fade from memory. It can be a long, frustrating road to get to this point, but after a few little victories the SEPG can expect to see this on the horizon.

> "Quietly making noise, [Ticking] off the old killjoys. Not too soft, not too loud, just enough to draw a crowd."
> – *Quietly Making Noise*[10]

The SEPG can make noise with the changes brought about by process improvement, but it is the entire organization that gets the fanfare for big improvements. The SEPG's role is to beg, cajole, educate, hand-hold, facilitate, monitor, and ease the organization along the journey to improvement. The old killjoys will definitely resist and get ticked off about changing the way things are done, but some will listen. As little victories are won, more will listen, and the noise will build. The key is to keep plowing straight ahead on the course, adapting to the changing conditions and gathering a crowd as you go. As the crowd builds, the momentum changes, and the little victories come closer together. This makes more and more noise until the crowd grows, and bigger victories are won. u

Acknowledgment

Thanks to Jimmy Buffett for the inspiration and insights. When the frustrating moments come along, and they will, I suggest quietly making a little noise with some of those kindergarten toys.

About the Author

Paul Kimmerly has 12 years experience in information systems development for the different incarnations of the Defense Finance and Accounting System (DFAS), Financial Systems Activity. For four years he has served as a member of the SEPG, and has been chairman of the group for the past two years. In addition to his local duties, he chaired a group that represented six sites within his organization's parent agency, the DFAS Financial Systems Organization (FSO). This corporate group addresses FSO-wide process improvement issues. He presented part of a tutorial entitled Transition Successes from the Field at the 1997 Software Engineering Symposium.

DFAS-FSAKC/KZ
1500 E. 95th Street
Kansas City, MO 64197
Voice: 816-926-5364 DSN 465-5364
Fax: 816-926-6969 DSN 465-6969
E-mail: pjkimmerly@cleveland.dfas.mil

Notes

1. Buffett, Jimmy, "Cowboy in the Jungle," Son of a Son of a Sailor, MCA, 1978.
2. Buffett, Jimmy, "Growing Older, But Not Up," Coconut Telegraph, MCA, 1980.
3. Buffett, Jimmy and Mac McAnally, "Changing Channels," Off to See the Lizard, MCA, 1986.
4. Buffett, Jimmy and Matt Benton, "Schoolboy Heart," Banana Wind, Margarittaville, 1996.
5. Buffett, Jimmy, Michael Utley, Will Jennings, and Josh Leo, "Who's the Blond Stranger," Riddles in the Sand, MCA, 1985.
6. Buffett, Jimmy and Michael Tschudin, "Quietly Making Noise," Fruitcakes, Margarittaville, 1994.
7. McAnally, Mac, "It's My Job," Coconut Telegraph, MCA, 1980.
8. Buffett, Jimmy, "Changes in Latitudes, Changes in Attitudes," Changes in Latitudes, Changes in Attitudes, MCA, 1977.
9. Buffett, Jimmy, "Wonder Why You Ever Go Home," Changes in Latitudes, Changes in Attitudes, MCA, 1977.
10. Buffett, Jimmy and Michael Tschudin, "Quietly Making Noise," Fruitcakes, Margarittaville, 1994.

CROSSTALK

Sponsor	***Lt. Col. Joe Jarzombek*** 801-777-2435 DSN 777-2435 jarzombj@software.hill.af.mil
Publisher	***Reuel S. Alder*** 801-777-2550 DSN 777-2550 publisher@stsc1.hill.af.mil
Managing Editor	***Forrest Brown*** 801-777-9239 DSN 777-9239 managing_editor@stsc1.hill.af.mil
Senior Editor	***Sandi Gaskin*** 801-777-9722 DSN 777-9722 senior_editor@stsc1.hill.af.mil
Graphics and Design	***Kent Hepworth*** 801-775-5555 ext. 3027 graphics@stsc1.hill.af.mil
Associate Editor	***Lorin J. May*** 801-775-5555 ext. 3026 backtalk@stsc1.hill.af.mil
Editorial Assistant	***Bonnie May*** 801-775-5555 ext. 3022 customer_service@stsc1.hill.af.mil
Features Coordinator	***Heather Winward*** 801-775-5555 ext. 3023 features@stsc1.hill.af.mil
Customer Service	***Barbara McDonald*** 801-777-8045 DSN 777-8045 custserv@software.hill.af.mil
Fax	801-777-8069 DSN 777-8069
STSC On-Line	http://www.stsc.hill.af.mil
***CROSSTALK* On-Line**	http://www.stsc.hill.af.mil/Crosstalk/crostalk.html
ESIP On-Line	http://www.esip.hill.af.mil

Subscriptions: Send correspondence concerning subscriptions and changes of address to the following address:

Ogden ALC/TISE
7278 Fourth Street
Hill AFB, UT 84056-5205

E-mail: custserv@software.hill.af.mil
Voice: 801-777-8045 DSN 777-8045
Fax: 801-777-8069 DSN 777-8069

Editorial Matters: Correspondence concerning *Letters to the Editor* or other editorial matters should be sent to the same address listed above to the attention of *CROSSTALK Editor* or send directly to the senior editor via the E-mail address also listed above.

Article Submissions: We welcome articles of interest to the defense software community. Articles must be approved by the *CROSSTALK* editorial board prior to publication. Please follow the *Guidelines for CROSSTALK Authors*, available upon request. We do not pay for submissions. Articles published in *CROSSTALK* remain the property of the authors and may be submitted to other publications.

Reprints and Permissions: Requests for reprints must be requested from the author or the copyright holder. Please coordinate your request with *CROSSTALK*.

Trademarks and Endorsements: All product names referenced in this issue are trademarks of their companies. The mention of a product or business in *CROSSTALK* does not constitute an endorsement by the Software Technology Support Center (STSC), the Department of Defense, or any other government agency. The opinions expressed represent the viewpoints of the authors and are not necessarily those of the Department of Defense.

Coming Events: We often list conferences, seminars, symposiums, etc., that are of interest to our readers. There is no fee for this service, but we must receive the information at least 90 days before registration. Send an announcement to the *CROSSTALK* Editorial Department.

STSC On-Line Services: STSC On-Line Services can be reached on the Internet. World Wide Web access is at http://www.stsc.hill.af.mil. The STSC maintains a Gopher server at gopher://gopher.stsc.hill.af.mil. Its ftp site may be reached at ftp://ftp.stsc.hill.af.mil. The Lynx browser or gopher server can also be reached using telnet at bbs.stsc.hill.af.mil or by modem at 801-774-6509 or DSN 775-3602. Call 801-777-7989 or DSN 777-7989 for assistance, or E-mail to portr@software.hill.af.mil.

Publications Available: The STSC provides various publications at no charge to the defense software community. Fill out the Request for STSC Services card in the center of this issue and mail or fax it to us. If the card is missing, call Customer Service at the numbers shown above, and we will send you a form or take your request by phone. The STSC sometimes has extra paper copies of back issues of *CROSSTALK* free of charge. If you would like a copy of the printed edition of this or another issue of *CROSSTALK*, or would like to subscribe, please contact the customer service address listed above.

The ***Software Technology Support Center*** was established at Ogden Air Logistics Center (AFMC) by Headquarters U.S. Air Force to help Air Force software organizations identify, evaluate, and adopt technologies that will improve the quality of their software products, their efficiency in producing them, and their ability to accurately predict the cost and schedule of their delivery. *CROSSTALK* is assembled, printed, and distributed by the Defense Automated Printing Service, Hill AFB, UT 84056. *CROSSTALK* is distributed without charge to individuals actively involved in the defense software development process.

Management 101

Are you the kind of manager who frequently sticks your neck out for your subordinates? Do your loyal, happy employees reach deadlines like clockwork? And last, are you some kind of masochist or what? Apparently you are, because your management style is just creating extra work for yourself, keeping you from fulfilling the number one management objective: getting promoted. So if you're the type of manager who gets sidetracked by quaint ideals like "making things work" or "getting results," it's time to revisit some management fundamentals.

Getting promoted – In many organizations, you can do this without having to produce any tangible deliverables—you just need to maintain the status quo while "networking" with some higher ups. However, in other organizations, you have to first make some shortsighted decisions that enhance your stature. But first, you'll need to assemble a team of empowered, capable employees—empowered to take the blame for your mistakes and capable of producing impressive-looking work for which you can take credit. And then you will have to give your managers the impression that your team is oozing from its orifices with ...

Productivity – If your project is on schedule and your employees are going home at a decent hour, that doesn't count as productivity. People don't look like they're working hard unless they're up against bleak odds—name one statue in honor of a leader who easily did what was promised. If the bleak odds don't already exist, you can create them: If your project should require eight months to complete, push the schedule ahead two months. This ensures that important steps will be cut early on to save time; for the remainder of the project your employees will then need to put in a lot of unpaid overtime redoing things, which raises your promotability stature above those "everyone-out-at-five" managers.

Meanwhile, you need to be spending time on the golf course with the right people to ensure that you're promoted before your project deadline. A year from now, a stooge needs to be solidly in your former position when the project, already four months late, self-implodes due to your early leadership.

Employee relations – Employees will whine on and on about their long hours and deteriorating family life without stopping once to consider the cost of an in-dash CD changer in the Porche you are planning to buy. But occasionally, an employee will present a legitimate concern that causes you to feel empathy and to accommodate those needs. Ignore this feeling. It leads to too much touchy-feely talk and wastes your time. It's much more efficient for you to respond with accusations of disloyalty and thinly veiled threats. When those tools don't work, give individual employees the impression they are just teetering on the brink of a promotion, so they'll think they need to stay on your good side.

Yes, employee empathy is for sissies, but to avoid mutiny you'll still need to pretend you care about their irrelevant personal lives. For example, just for the heck of it, agree to look the other way if an employee takes an extra-long lunch to attend the funeral of a spouse. Or occasionally accommodate a worker's unique circumstances, such as personally giving her a top-of-the-line laptop computer to work on between contractions while she's in the delivery room.

Employee rewards and compensation – Merit-based reward systems inspire the wrong behavior—those goody-two-shoes "self-motivated" employees can make life so hard. You can drive most of these types away with inadequate pay, leaving you with a foundation of yes-men and yes-women who would never distract you with ideas and "improvements" that draw unwanted attention to your project.

However, there will always be a few "movers and shakers" working for you. They'll provide cohesion and expertise to the team, often providing heroic efforts that boost your project's status within the organization. These people must be crushed. You don't want to spend months intentionally missing two-foot putts only to have some goody-goody obtain the favor of a "results-driven" manager.

Of course, you could just try to move up by providing superior long-term results, but then where's the fun? And what would your employees do if they had free time anyway? Have a social life? Fall in love and get married? You'd better order some extra laptops for those honeymooners. – Lorin May

Got an idea for BACKTALK? Send an E-mail to backtalk@stsc1.hill.af.mil