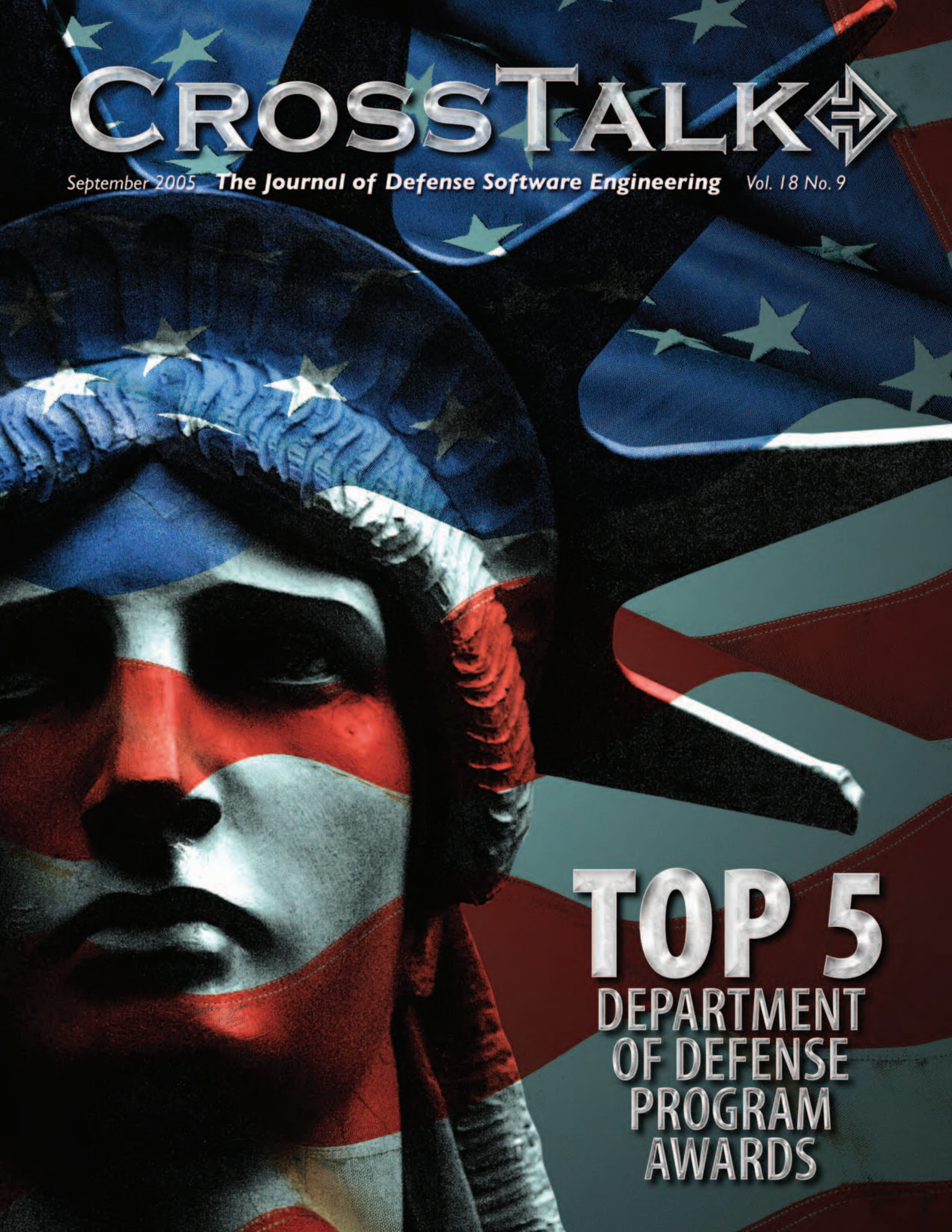
CrossTalk

September 2005 **The Journal of Defense Software Engineering** Vol. 18 No. 9



TOP 5 DEPARTMENT OF DEFENSE PROGRAM AWARDS

Report Documentation Page

Form Approved
OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **SEP 2005**	2. REPORT TYPE	3. DATES COVERED **00-00-2005 to 00-00-2005**

4. TITLE AND SUBTITLE **CrossTalk: The Journal of Defense Software Engineering. Volume 18, Number 9, September 2005**	5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **OO-ALC/MASE,6022 Fir Ave,Hill AFB,UT,84056-5820**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S) 11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**32**	

Top Five Department of Defense Program Awards

Best Practices

Software Engineering Technology

ON THE COVER
Cover Design by Kent Bingham.

Departments

CROSSTALK

76 SMXG Co-Sponsor *Kevin Stamey*
309 SMXG Co-Sponsor *Randy Hill*
402 SMXG Co-Sponsor *Tom Christian*
Publisher *Tracy Stauder*
Associate Publisher *Elizabeth Starrett*
Managing Editor *Pamela Palmer*
Associate Editor *Chelene Fortier-Lozancich*
Article Coordinator *Nicole Kentta*
Creative Services Coordinator *Janna Kay Jensen*
Phone (801) 775-5555
Fax (801) 777-8069
E-mail crosstalk.staff@hill.af.mil
CrossTalk Online www.stsc.hill.af.mil/crosstalk

Oklahoma City-Air Logistics Center (76 SMXG), Ogden-Air Logistics Center (309 SMXG), and Warner Robins-Air Logistics Center (402 SMXG) Software Maintenance Groups (SMXG) are the official co-sponsors of CrossTalk, The Journal of Defense Software Engineering. The SMXG's and the Software Technology Support Center (STSC) are working jointly to encourage the engineering development of software to improve the reliability, sustainability, and responsiveness of our warfighting capability.

The STSC is the publisher of CrossTalk, providing both editorial oversight and technical review of the journal.

S T S C

Subscriptions: Send correspondence concerning subscriptions and changes of address to the following address. You may e-mail us or use the form on p. 30.

309 SMXG/MXDB
6022 Fir AVE
BLDG 1238
Hill AFB, UT 84056-5820
(801) 775-5555

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 Author Guidelines, available at <www.stsc.hill.af.mil/crosstalk/xtlkguid.pdf>. CrossTalk does not pay for submissions. Articles published in CrossTalk remain the property of the authors and may be submitted to other publications.

Reprints: Permission to reprint or post articles must be requested from the author or the copyright holder and coordinated with CrossTalk.

Trademarks and Endorsements: This Department of Defense (DoD) journal is an authorized publication for members of the DoD . Contents of CrossTalk are not necessarily the official views of, or endorsed by, the U.S. government, the DoD, or the STSC. All product names referenced in this issue are trademarks of their companies.

Coming Events: Please submit conferences, seminars, symposiums, etc. that are of interest to our readers at least 90 days before registration. Mail or e-mail announcements to us.

CrossTalk Online Services: See <www.stsc.hill.af.mil/crosstalk>, call (801) 777-7026, or e-mail <stsc.webmaster@hill.af.mil>.

Back Issues Available: Please phone or e-mail us to see if back issues are available free of charge.

Integrated Teams and Sound Processes Bring Success

This month, CrossTalk is pleased to publish articles from the five winning programs of the Top 5 Department of Defense Program Awards 2004 contest. As one of the judges on this year's selection committee, I was delighted to see that many of the nominees were fine examples of projects that are succeeding at developing, sustaining, or integrating software into government systems today. Too often we hear of projects that are failing. It's refreshing to see projects that are triumphant given continual advances in software technology along with the challenging goals of producing quality software faster, better, and cheaper.

A common theme among the Top 5 winners is having an integrated team that continually involves its stakeholders. Success can be realized when customers, acquirers, developers, testers, configuration managers, end users, etc. all work together throughout the system life cycle, especially during requirements analysis.

Following sound processes is another key factor to this year's Top 5 winners' success. Whether it means employing a spiral development approach, or agile programming practices, or Capability Maturity Model Integration practices, a common thread is to implement and follow well-defined processes to ensure quality and customer satisfaction. Congratulations to all those involved in these programs.

Continuing on this theme of developing software with proven practices, a caution to involve management in process implementation is discussed in *The Myth of the Best Practices Silver Bullet* by Michael W. Evans, Corinne Segura, and Frank Doherty. These authors discuss why it is critical that management select the organization's best practices and understand the costs and impacts involved; otherwise, practices may not be followed and can instead be harmful to a project.

Next, Joe H. Lindley brings us *Measure Like a Fighter Pilot*. Learn how measurement analysts can benefit from a focused strategy such as the Observe, Orient, Decide, and Act Loop. In *Applying Functional TSP to a Maintenance Project*, Ellen George and Dr. Steve Janiszewski share their success in employing both the Personal Software Process and Team Software Process in maintenance project environments. Finally, David P. Quinn discusses why organizations need to be careful when using project measures as a means of rewarding employee good performance in *Tying Project Measures to Performance Incentives*.

As software acquisition, development, and sustainment in the Department of Defense continues to challenge us, I hope this month's set of articles provides helpful information as your team strives to succeed and meet customer needs.

Top 5 Department of Defense Program Awards 2004

The Office of the Under Secretary of Defense and the National Defense Industrial Association present the Top 5 Department of Defense Program Awards 2004. These programs were chosen for their excellence and success at using well-defined and proven processes to develop, manage, and integrate software into deliverable systems. The programs are listed alphabetically and do not indicate a place order.

- **Lightweight Handheld Mortar Ballistic Computer**
  Service: U.S. Army
  Industry Contractor: None
- **Marine Corps Total Force System**
  Agency: Technology Services Organization, Defense Finance and Accounting Service
  Industry Contractor: Computer Sciences Corporation
- **Near Imaging Field Tower Implementation**
  Agency: Naval Research Laboratory Midway Research Center
  Industry Contractor: Joint effort between Mnemonics, Inc., Assurance Technology Corp., Harris Corp., Blaseware, Analex, and SAIC
- **SmartCam3D System**
  Agency: NASA
  Industry Contractor: Rapid Imaging Software
- **Warfighter's Simulation**
  Service: Program Executive Office-Simulation, Training, and Instrumentation, U.S. Army
  Industry Contractor: Lockheed Martin Simulation Training and Support

Tracy L. Stauder

Tracy L. Stauder
Publisher

Lightweight Handheld Mortar Ballistic Computer

Mike Patriarca
Project Manager Mortars

Mark Zhelesnik
Armament Research Development and Engineering Center

While supporting Operation Iraqi Freedom, the enhanced functionality of the Lightweight Handheld Mortar Ballistic Computer (LHMBC) in providing updated ballistic solutions to fire missions for infantry mortar combat units has proven to be a great success. With the LHMBC, mortar fire direction centers are computing faster, more accurate ballistic solutions to fire missions for 60mm, 81mm and 120mm mortar systems.

Today's fast-paced battlefield requires responsive and accurate weapon platforms to stay ahead of enemy forces. The XM 32 Lightweight Handheld Mortar Ballistic Computer (LHMBC) incorporates advances in digital electronics and software capabilities that make it a superior fire control system. Jointly, the product manager for Mortar Systems (PM Mortars) and the U.S. Army Research and Development Center (ARDEC) located at Picatinny, N.J., have developed and are fielding the LHMBC.

The LHMBC provides the soldier, for the first time, a lightweight handheld fire control system with an integrated Global Positioning System (GPS) to determine his/her location, and a modem that allows for digital communication within the fire support network. The system calculates ballistic solutions to fire missions and provides fire support coordination measures with functionality, including digital missions (Grid, Polar, Shift, Adjust Fire, Fire For Effect, Immediate Suppression, Immediate Smoke, Precision Registration) with methods of control (At My Command, When Ready), six simultaneous missions, three final protective fires, priority targets, fratricide checks, check fire, single safety fan with multiple segments, digital meteorological data, and digital variable message format messages to/from the Advanced Field Artillery Tactical Data System and the Forward Observer System.

This software-intensive LHMBC program makes use of corporate software and system engineering methodologies and processes that have led to the successful accomplishment of several key milestones, including urgent materiel release and fielding of 36 LHMBC systems. The Urgent Materiel Release of LHMBC Vers. 1.1 to 3/2 Stryker Brigade Combat Team was approved in August 2004 to satisfy an urgent U.S. Army requirement in support of Operation Iraqi Freedom.

The LHMBC urgent fielding effort was particularly unique. To meet accelerated fielding, PM Mortars used ARDEC as a central location to load software onto a host computer, assemble all hardware components into kits, and stage all fire control equipment for delivery. The program strategy calls for fielding incremental functionality improvements with two additional releases scheduled for fielding in the third quarter of 2005 and fiscal year 2006. Incremental development continues through fiscal year 2006 with planned fielding through fiscal year 2008.

> *"The LHMBC provides the soldier, for the first time, a lightweight handheld fire control system with an integrated Global Positioning System (GPS) to determine his/her location, and a modem that allows for digital communication within the fire support network."*

This incremental fielding strategy best meets the user's urgent requirement. The Incremental Development Life-Cycle Model dovetails with this strategy, thus it was the model of choice for the LHMBC software development team. During the planning phase functionality was prioritized by the user. This input was used by the PM to determine the functionality of each software version.

LHMBC Development

LHMBC is one of several spin-offs of the Mortar Fire Control System (MFCS), also developed in a joint effort between PM Mortars and ARDEC. Initially fielded in fiscal year 2003, the MFCS program was selected for an Army's research and development award in 2003; its architecture is covered by a government patent (pending). Leveraging the flexible MFCS architecture and reusing approximately 80,000 lines of MFCS code (40 percent of the total LHMBC code) resulted in an estimated cost savings of $2.4 million and a schedule savings of 18 months for the LHMBC program. The system also uses the Mortar Ballistic Kernel that is common to both MFCS and LHMBC (an additional 40,000 lines of reused code).

The LHMBC software application program is hosted on a Ruggedized Personal Digital Assistant (R-PDA) that replaces the aging M23 Mortar Ballistic Computer. The R-PDA hardware is basically a commercial off-the-shelf device acquired through the PM Common Hardware and Software (CHS) located at Ft. Monmouth, N.J. Choosing available hardware saved development time, maintained commonality of fielded hardware, and reduced cost by utilizing economies of scale in quantities purchased through PM CHS. The hardware is produced by Tallahassee Technology Incorporation of

® Capability Maturity Model and CMMI are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

Florida and provided through the PM CHS contract.

The LHMBC incorporates an innovative fire control software application that provides enhanced capability to the mortar operator by reducing response times; providing enhanced accuracy; and allowing the mortar to be more mobile, responsive, and lethal. This advanced fire control system provides effective means to rapidly disseminate relevant data among weapon and command-and-control systems. The LHMBC provides unique capabilities available for the first time ever on a R-PDA, with a significant weight reduction from eight pounds to three pounds at one-eighth the size of previously fielded lightweight fire control systems.

The ability to quickly react to an urgent need and the continuing success of this program is due to the well-defined proven practices and processes used to develop, manage, and integrate the software into a state-of-the-art R-PDA computer. The in-house software development and system integration team used tools such as Lean Six Sigma processes and procedures, earned value management, and the Software Engineering Institute's (SEI) Capability Maturity Model® Integration (CMMI®) Level 3 software development process to enhance overall efficiencies and quality of the software.

The Build Team and Processes

The LHMBC integrated product team (IPT) consisted of in-house members along with members from the test and user communities, including actual soldiers who operate the system. The test and user communities were involved early during system development to address any concerns or issues and reduce the number of potential changes and rework required.

The user provided valuable feedback on screen layouts, content, and sequencing of messages to process missions. To ensure the soldiers have a reliable product, intensive qualification testing was performed in the Systems Integration Lab, including formal qualification testing and an initial operational test. Furthermore, an independent validation and verification (IV&V) process was also employed prior to the product baseline. During the IV&V process, physical and functional configuration audits were done on the software.

The industry- and government-sponsored CMMI process improvement model provides a set of best practices that address product quality, productivity, performance, cost, and stakeholder satisfaction. CMMI Level 3 compliance ensures that all CMMI-defined process areas at Levels 2 and 3 are implemented. CMMI Level 3 processes are addressed in the following areas: requirements development, technical solutions, product integration, verification, validation, integrated project management, risk management, decision analysis and resolution, and integrated supplier management. The ARDEC effectively implemented these process improvement efforts in the development of the LHMBC.

The U.S. Army's XM 32 Lightweight Handheld Mortar Ballistic Computer is a significant improvement over the M23 Mortar Ballistic Computer – its predecessor – because it is lighter, provides digital connectivity to the fire support network, processes ballistic solutions faster, and provides system location through an integrated Global Positioning System.

The ARDEC Armament Software Engineering Center (SEC) has experienced success in implementing CMMI Level 3-compliant processes for software-intensive system projects such as the LHMBC. This success has led to the current process improvement initiative that resulted in upgrading the ARDEC Armament SEC's Organizational Standard Process to CMMI Level 5 compliance. The LHMBC and other current software-intensive system developments undertaken at ARDEC Armament SEC now implement the CMMI Level 5-compliant Organizational Standard Process.

LHMBC and the other Level 5-compliant development efforts are scheduled to be appraised in the fall of 2005 by SEI approved appraisers. Software development and process tools such as Embedded Visual Studio for C++, Visual SourceSafe, and processMax were utilized for the LHMBC software development process. These tools assisted the developers and the IPT members with data management, configuration management, and process management.

The LHMBC is a significant improvement over the M23 Mortar Ballistic Computer because it is lighter, provides digital connectivity to the fire support network, processes ballistic solutions faster, and provides system location through GPS. The combined efforts of the PM Mortars and ARDEC teams, along with support from our test and user community, has led to the successful development of the infantry's first fully digitized lightweight mortar fire control system, thus making the infantry mortarman an integral part of the digitized battlefield.◆

Authors/Project Points of Contact

Lt. Col. Andre C. Kirnes
SFAE-AMO-CAS-MS
BLDG 162 South
Picatinny Arsenal, NJ 07806-5000
Phone: (973) 724-4209
Fax: (973) 724-7266
E-mail: andre.kirnes@us.army.mil

Mike Patriarca
SFAE-AMO-CAS-MS
BLDG 162 South
Picatinny Arsenal, NJ 07806-5000
Phone: (973) 724-3571
Fax: (973) 724-7266
E-mail: mpatria@pica.army.mil

U.S. Marines – First Into Battle and First With a Unique Pay and Personnel System

Jimmy W. Selph
Computer Sciences Corporation

The Marine Corps Total Force System (MCTFS) is the only integrated military pay and personnel system in the Department of Defense (DoD). MCTFS manages more than 498,000 Marine records for active, reserve, and retired members; processes in excess of 17 million transactions yearly; and computes an average gross payroll of $238 million per semi-monthly pay period totaling $5.712 billion in payments annually. MCTFS provides added value to the customer's mission – paying service members accurately and on time. Cumulative for fiscal year 2004, MCTFS demonstrated that it paid all active duty and reserve Marines on time with 99.92 percent and 99.83 percent pay accuracy, respectively. To date for fiscal year 2005, the accuracy is 100 percent for both active duty and reserve Marines. The positive emotional and financial impact of paying deployed Marines accurately and on time is especially noteworthy to the commanders in the field in terms of unit morale and welfare.

The U.S. Marine Corps (USMC) is one tough customer with demanding expectations, especially when it relates to the Marine Corps Total Force System (MCTFS), the USMC's integrated military pay and personnel system. MCTFS manages more than 498,000 Marine records for active, reserve, and retired members; processes in excess of 17 million transactions yearly; and computes an average gross payroll of $238 million per semi-monthly pay period, totaling $5.712 billion in payments annually. Most importantly, MCTFS needs to pay service members accurately and on time.

The Kansas City Technology Services Organization (TSO-KC) and their prime contractor, Computer Sciences Corporation, are dedicated to ensuring continuous, responsive, and effective military pay and personnel information technology support. The technical managers of MCTFS clearly understood the task at hand when the commandant of the Marine Corps stated:

> We remain committed to transforming our manpower processes by leveraging the unique capabilities resident in the Marine Corps Total Force System, our fully integrated personnel, pay, and manpower system that serves active, reserve, and retired members … Additionally, MCTFS facilitates our single source of manpower data, directly feeding our Operational Data Store Enterprise and Total Force Data Warehouse. This distinctive capability allows us to accurately forecast manpower trends and fuels our Manpower Performance Indicators, which provide near real-time graphical representation of the Corps' manpower status such as our deployment tempo. [1]

The Unique Pay and Personnel System

MCTFS is jointly sponsored and funded by the Defense Finance and Accounting Service (DFAS) and the USMC. It is the only integrated military pay and personnel system in the Department of Defense (DoD). *System integration is a major contributing factor* to successful management and administration of pay and personnel functions for the DFAS and the USMC. *Having management from these two organizations co-located with the system developer* (TSO-KC) *has proven to be a key component* of MCTFS success.

System and business integration of these two domains (pay and personnel) means that from a single source, one event can be reported and MCTFS will properly update both domains. A simplified example of single source reporting is when a Marine is promoted in rank from corporal to sergeant. System processing of the promotion transaction within MCTFS encompasses all the required programming logic to ensure both pay and personnel information are concurrently updated by the single input promotion transaction.

The other components of the armed services utilize separate information systems to manage pay and personnel functions and, therefore, must input multiple, and by definition duplicate, transactions to report an event such as a promotion in rank. Integration of pay and personnel and single source reporting is unique to MCTFS and yields tangible and intangible benefits. These benefits include fewer resources to perform simplified input reporting procedures, seamless integration of pay and personnel functions, and no synchronization problems between disparate pay and personnel systems.

MCTFS is the single, authoritative data source and system of record for all pay and personnel information for Marine service members. Input MCTFS transactions are generated from a variety of net-centric ancillary applications that include stand-alone, client/server, and Web-based systems. The stand-alone systems such as Remote Access Pay Transaction Reporting System allow users to create transactions while not connected to a network in a remote deployed environment. These transactions can then be submitted into MCTFS once connectivity is available.

Client/server systems such as the Unit Diary/Marine Integrated Personnel System allow users to work in a Wide Area Network environment, share data, and pool transactions with other system users.

The award-winning[1] MCTFS front-end Web-based system Total Force Administration System (TFAS)/Marine OnLine provides users the ability to generate self-service transactions. Transactions originating from these ancillary systems are processed by MCTFS in conjunction with other transactions, and are subsequently available for querying by users in the Operational Data Store enterprise (ODSe), an Oracle 10g relational database. This database is optimized to support the entire Marine Corps and the breadth of process stakeholders and decision makers.

Founded on a strict set of business rules, MCTFS is architected on the premise that Marine commanding officers are accountable and responsible for the personnel and pay management of Marines. MCTFS stores data that includes operations and deployment dates, promotions, performance evaluations, duty-station assignments, personal awards, reserve drills, skills and occupations, and training/education information. MCTFS contains data to correctly pay every Marine with regard to state/federal income taxes, residency information, entitlements and allowances, special incentive pay, and allotments. Manpower, personnel, pay, and training data are readily available via the ODSe component in the form of preformatted reports or via commercial off-the-shelf software such as Cognos Impromptu and Powerplay tools.

Software Engineering Maturity

Software quality assurance and organizational institutionalization of software engineering practices are keys to ensuring timely and accurate Marine pay and personnel administration. MCTFS achieved the Software Engineering Institute's (SEI) Capability Maturity Model® for Software (SW-CMM®) Level 2 rating in 1997, and SW-CMM Level 3 rating in 2000. The key process areas associated with Level 2 and Level 3 are thoroughly institutionalized. MCTFS has institutionalized processes consistent with SW-CMM Level 4, and was informally assessed at SW-CMM Level 4 in May 2004 by an SEI Authorized Lead Appraiser. A formal SW-CMM Level 4 assessment was conducted in June 2005[2], even while plans are afoot to transition to CMM Integration℠.

TSO-KC subjects its family of systems to annual benchmarks conducted by Gartner, Inc. <www.gartner.com>. Benchmarking conducted by Gartner, Inc. captures and stores information technology performance metrics then compares those TSO metrics to metrics obtained from similar participating organizations.

Since 1997, the average defect rate per 1,000 function points (FPs) for MCTFS is 1.76. This low average defect rate consistently ranks MCTFS in the top quartile of its software industry database. The quality of the MCTFS software releases continues to exceed expectations based on empirical data.

TSO-KC managers attribute much of the success to the institutionalized software process improvement activities that ensure managerial and engineering functions are standardized, defined, and documented throughout the life cycle. The following highlights are from the most current Gartner benchmark for MCTFS:

The Total Force System winning team included, from left, Director Clinton L. Swett, Defense Finance and Accounting Service, Technology Services Organization; Chief Warrant Officer Harry Sanchez, U.S. Marine Corps Manpower and Reserve Affairs; Robert Brown, U.S. Marine Corps Manpower and Reserve Affairs; and Director Gary Hayes, Kansas City Operations, Computer Sciences Corporation.

- FPs supported per person at 4,623 is 89 percent better than government peer group (2,440 FP/full-time employees).
- Cost per FP supported at $33 is 8 percent lower than the government peer group.
- Cost per FP developed at $168 is 54 percent lower than the database average and 81 percent lower than the government peer group.
- The 903 FPs developed per person is five times higher than the government peer group.
- Quality levels, as measured by defects per 1,000 FPs (slightly more than five, including cosmetic defects), remain lower (better) than average for support.
- The complexity of the business environment in development is higher than the government peer average for both enhancements and new development.

The USMC, TSO-KC, and their prime contractor Computer Sciences Corporation are proud of their tradition of cooperation in managing MCTFS and its family of integrated applications. MCTFS today represents the government's and industry's leading practices, hardware, software, and programming languages. MCTFS will meet the Marine Corps' needs into the next decade and beyond: the system is sufficiently scalable and adaptable to provide capabilities for enterprise-wide use within the DoD.◆

Reference

1. Hagee, Gen. Michael W. Senate Armed Services Committee, 10 Feb. 2005.

Note

1. TFAS was awarded the 2003 Department of Defense Chief Information Officer Award, the 2003 Department of the Navy e-Government Award, and the 2003 Government Computer News Innovation in Technology Award.
2. The assessment resulted in MCTFS achieving SW-CMM Level 4.

Project Points of Contact

Clinton L. Swett
Defense Finance and
Accounting Service
1500 E 95th ST
Kansas City, MO 64197
Phone: (816) 926-7373
Fax: (816) 926-3361
E-mail: clinton.l.swett@dfas.mil

Gary Hayes
Computer Sciences Corporation
1500 E 95th ST
Kansas City, MO 64197
Phone: (816) 926-2393
Fax: (816) 926-8049
E-mail: ghayes@csc.com

[SM] CMM Integration is a service mark of Carnegie Mellon University.
[®] CMM is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

A NIFTI Solution to Far-Field Antenna Transformations

Windie Borodin and Danielle King
Midway Research Center

The Near Imaging Field Tower Implementation was developed to provide a mid-range system that performs far-field characterization of several 60-foot antenna systems. The challenge was to take a highly complex development effort and bring it in on time and within cost. This article describes the team's success using an object-oriented design under a tailored IBM Rational Unified Process and established metrics to monitor the progress.

The mission of the Midway Research Center (MRC) located in Stafford, Va., is to function as a high-precision signal source for calibrating and testing national and tactical systems. Among its numerous assets are three antennas 60-foot in diameter that provide highly accurate radio frequency (RF) signals. Due to their size and location, they require a system to characterize the antennas to be located as much as 25 miles away from where they are actually located. This type of system requires aircraft to accomplish this task, making it economically impossible to characterize the antennas on a regular basis.

The U.S. Navy decided to develop a mid-range calibration system approximately 884 feet away from the antennas to transform the RF data from the antenna into a pattern that makes them appear 25 miles away. This is known as the Near Imaging Field Tower Implementation (NIFTI) project. The goal was to produce an operational system that could be used on a regular basis to collect data from the test signal being transmitted from the antennas. By analyzing this data, the system would be able to determine what factors are needed to correct the actual signal being transmitted. This would produce a better quality and higher precision signal being transmitted to the desired targets.

The system was developed and tested using the Antenna Tracking Subsystem (ATS)-3 antenna. This antenna scans across a reflector that is mounted at the calibration tower. Data is collected as the ATS-3 antenna generates a continuous wave tone and scans across the calibration tower. This data is then transformed into a far-field pattern using a Fourier transform technique, a method for analyzing periodic functions.

NIFTI Team

The core development team was located on site with the system user and was comprised of both government and contractor personnel. Within the development, tasks were distributed among numerous contractors, including Mnemonics, Inc.; Assurance Technology Corporation; Harris Corporation; Blazeware; Analex; and Science Applications International Corporation. The project overcame challenges inherent when different companies work side-by-side such as corporate alliances and company policies. It also tackled the required knowledge of both RF theory and software engineering.

Those chosen for the task ranged in experience from interns to Ph.D.s. Expertise in different areas was shared among team members. Junior level software engineers trained senior level personnel on programming best practices and software engineering principles, while the senior level personnel trained others on RF theory. This coordination provided for maximum use of resources and overall team strength.

Developing alongside the ultimate system user also aided project success. The NIFTI team established processes (i.e., document and code reviews, risk management, configuration management, and training) that engaged the customer (U.S. Navy) and the end user (MRC operators) in the design and development process. These processes required the end user to be an active participant in the development effort. The end user participated in all document reviews, user interface working groups, all system integration, and acceptance testing.

The project team established a training process with the end-user software maintenance group where the software maintainers became part of the development team to learn and understand the software before it was delivered. The end user also held a seat on the Configuration Control Board and was able to generate change requests against the system at any time during the development process. This allowed the end user and customer the ability to provide input into the design and determine what changes were important.

Before this effort began, the end user was trained along with the developers on any new tool or process that was being used during the development. The team collected metrics on process, performance, change requests, budget, and schedule. These were provided at every design review to the customer and end user. This kept the customer informed about the project's progress. It also established a close working relationship, which allowed for the end user/customer to be involved in every major step of the development process.

Software Development

The NIFTI project was developed using object-oriented design and a tailored IBM Rational Unified Process. A development suite was established consisting of the Rational Suite of software that included ClearCase, ClearQuest, Requisite Pro, Rose, Test Manager, and McCabe Quality Toolset. A combination of UNIX (Sparc processors) and Windows (PC)-based servers were used to house the development environment.

The project followed the inception, elaboration, construction, and transition phases. In the inception phase, the budget and high-level schedule were established and the requirements were analyzed. As part of performing requirements analysis, all stakeholders met to come to an understanding of the requirements. Once this was accomplished, the requirements were then placed into Requisite Pro (a requirements tracking tool). Each requirement was assigned attributes such as (1) build that the requirement was to be implemented, (2) asset for which the requirement was needed, (3) use case that contained the requirement, and (4) test case it was assigned. Also contained in the database was an interpretation of the requirement that clearly explained the intent, which was agreed to by all stakeholders.

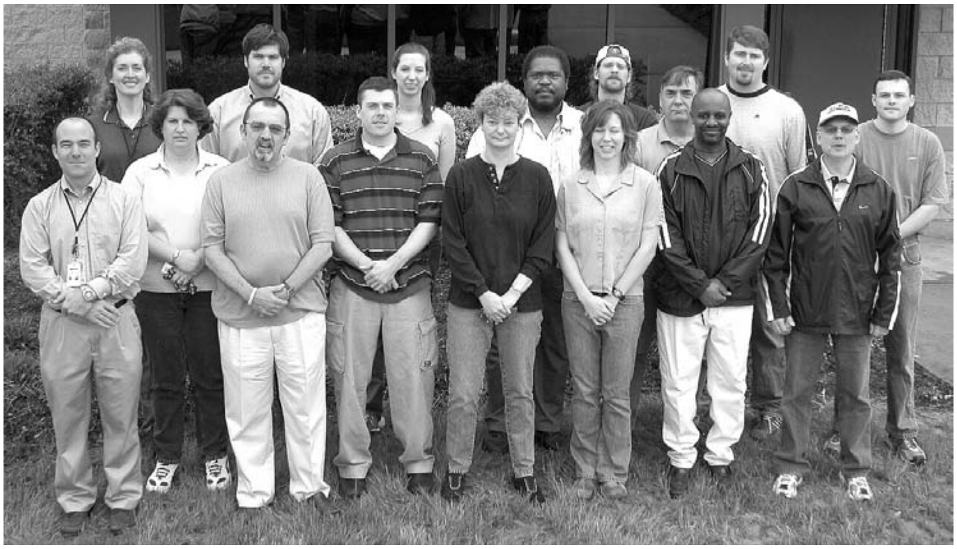
The end of the inception phase was designated by a review to the customer and end user. At this review, the team presented the project overview, system concept, external boundaries of the system being designed, software standards, requirements, acceptance criteria and verification matrix, and project risks.

For the elaboration and construction phases, an iteration plan was developed along with a schedule for that phase. The iteration plan contained what the team was going to accomplish during that phase along with evaluation and exit criteria, and the requirements that were to be addressed. At the end of each phase, the team provided a review to the customer and end user. During the elaboration phase, the team focused on prototyping some of the high-risk areas in the software. The main items presented during the end-of-phase review were system overviews, prototype results, use cases, test plans, refined software estimates, defined architectures, requirements mapped to use cases, requirements mapped to construction iterations, structured software models, schedules, and risks.

The construction phase was divided into three builds, with each build being four to six months in duration. One obstacle the project faced was that the NIFTI system was being installed into an operational system, thus the system was not always available for testing some requirements. When this occurred, the requirement was deferred until the next build and a requirement initially slated for a later build was implemented in its place. During the build, the tester conducted usability tests with the end user that gave him the opportunity to use the system. Feedback was given to the developers so that required changes could be made. This helped produce a system that was user-friendly and met the end user's needs.

The reviews at the end of each build were assessments of how the project performed during that build and whether the project accomplished the goals contained within the iteration plan. It also included a presentation of the metrics gathered not only during the build, but also during the project life cycle. During the end-of-build presentation, a demonstration of the system was also given to the customer. The unique thing about the demo was that it was run by the end user. This demonstrated to the customer that the end user was part of the team and that their needs were being addressed. Also presented at the review were the plans for the next build along with any updated schedule or project risks.

The transition phase consisted of a Developers' Test and Evaluation and formal installation of the system into the operational asset. The testing was witnessed and signed off on by personnel from the site's Systems Engineering Department; also present were personnel from the operations staff – the eventual system user.



Members of the Near Imaging Field Tower Implementation team are, from left, (front row) Steve Rugari, Pauletta Morrast, Warren Wagner, Greg Dye, Windie Borodin, Alice Howell, Terry Arbnot, Tom Shea, (back row) Susan Dilbeck, Norman Byrne, Danielle King, George Williams, Mark Lucas, Mike Whitelaw, Josh Howell, and Jim Dye.

Project Monitoring

The NIFTI project was monitored by using these established metrics:

1. **Cost measurements.** Within budget.
2. **Schedule performance.** Within an 18-month development cycle.
3. **Assessment measurement.** Produced 12 source lines of code (SLOC) per day versus the seven SLOC industry standard.
4. **Open/closed convergence of defects.** Number of open defects converged with closed defects.
5. **Hours expended per defect.** Most defects were repaired in less than three hours.
6. **SLOC per defect.** Most defects were repaired in less than 10 lines of code.
7. **Defects per subsystem inspection and testing (I&T) versus system I&T.** Most defects were detected during subsystem I&T before being delivered to the testers.

The team also monitored the software to evaluate it in terms of complexity (all modules had a complexity of 10 or less using McCabe Tools), to check for memory leaks, to determine bottlenecks, and to test path coverage.

Summary

The innovative processes that incorporated the end user and customer throughout the software development process have proven to be key in achieving project success. These structured processes, risk mitigation, metrics, and close communication with the end user/customer have been instrumental in producing a quality product on time and within budget that met its requirements. The diverse backgrounds of the development team also proved to be instrumental in the success of the project.

Major benefits from the system were seen even before it was officially turned over to operations: It has been used to detect a feed offset and bore sight problem in the ATS', and to find a software problem that caused antenna tracking to be off in customer tests. The clients were grateful that these and other problems could now be identified and resolved much faster and easier than before.◆

Project Points of Contact

Windie Borodin
Midway Research Center
Naval Research Laboratory
635 Telegraph RD
Stafford, VA 22554
Phone: (703) 551-1992
Fax: (703) 221-3317
E-mail: borodin@kingcrab.nrl.navy.mil

Danielle King
Midway Research Center
Naval Research Laboratory
635 Telegraph RD
Stafford, VA 22554
Phone (703) 551-1975
Fax: (703) 221-3317
E-mail: king@kingcrab.nrl.navy.mil

SmartCam3D Provides New Levels of Situation Awareness

Frank Delgado
NASA

Mike Abernathy
Rapid Imaging Software

Janis White
Statistical Consulting

SmartCam3D (SC3D) is a hybrid synthetic vision system that combines live sensor information with data from a synthetic vision system to create a virtual cockpit window. This combination of technologies allows the system to circumvent some of the limitations from each approach. SC3D is supporting various Department of Defense, NASA, and commercial industry endeavors related to the operation of tele-operated and windowless cockpit designed vehicles.

The pertinent information related to the environment and operational envelope of a vehicle is commonly referred to as situation awareness (SA). A lack of SA is responsible for numerous aircraft accidents that claim hundreds of fatalities per year. Additionally, the monetary cost associated with poor SA is in the billions of dollars due to lost aircraft, increased training time, decreased capabilities, and operational inefficiencies. The Department of Defense (DoD), NASA, and the general aviation industry have all lost important assets because an operator did not have adequate SA.

The development of a visualization system that would greatly improve a pilot's SA would prove very valuable to both government and commercial organizations. SmartCam3D (SC3D) was developed to serve this function and has provided SA to pilots at levels not previously possible.

Development of SC3D principles and technology components began in 1996 as parallel efforts by Rapid Imaging Software (RIS) and NASA. RIS was developing a commercial product that would allow for the visualization of terrain. At the same time, NASA's Johnson Space Center was developing visualization technologies that would be used on the X-38 program. The X-38 program was a technology development program that was building a series of flight-test vehicles to determine what would be required to develop a fully functional crew return vehicle for the International Space Station. In 1997 both organizations developed a collaborative partnership to develop an advanced visualization system that would greatly enhance a pilot's SA.

A hybrid synthetic vision system (HSVS) combines live sensor data with information from a synthetic vision system (SVS) to create real-time, information-rich visuals. An SVS provides a computer representation of the operational environment and is typically created using digital elevation data, imagery, maps, charts, vehicle models, etc.

SVS visuals are typically full of information and can be used during day, night, or low-visibility conditions. An SVS by itself can suffer from *data freshness* problems. These problems can occur because the SVS visuals are created from data typically collected by a satellite or aircraft flying overhead, meaning the data could have been collected days, weeks, months, or years ago. In a dynamic environment, the information could be outdated and provide obsolete or inaccurate scene information.

A live sensor system can provide up-to-the-second information for the region of interest, but by itself is not very useful when visibility conditions are hampered by rain, snow, sand, fog, and smoke, which are frequently encountered by vehicles.

HSVS visuals allow the user to circumvent some of the limitations from each separate approach by providing real-time information-rich visuals. SC3D has become the government's premier HSVS system. The SC3D system, with its innovative concepts and one-of-a-kind software technology, blurs the traditional distinction between visual and instrumented flights/operations.

Strategies and Methodologies

The team organization played an important factor in the overall success of the SC3D. This team included experts with hands-on experience in software development, human factors, visualization, configuration management, avionics, and aviation piloting. Other major factors that played a significant role in the success of the SC3D system were the processes that were followed and the design decisions made during the planning phase.

The system was developed using a spiral development approach implementing agile programming practices, object-oriented design (OOD), and object-oriented programming (OOP). The team also followed a rigorous quality assurance program throughout the entire development, implementation, and deployment life cycle. Spiral development has allowed users to see an increasingly more functional system throughout the development of the SC3D system. It also allowed for an iterative approach to requirements development by providing successive spiral cycles that contained the lessons learned from previous spiral cycles. Agile programming permitted the developer to incorporate user feedback into the software within each spiral cycle. The OOD and OOP principles simplified maintenance, reuse, and capability augmentation of the core technologies.

Any problems encountered could be quickly isolated to individual components making problem identification and resolution much easier. As new capabilities were needed, the various components were easily created/modified. Although the SC3D system has more features than any visualization system currently available and supports real-time visualization of very complex environments, reliability has always been the most important factor. An intense quality assurance process was utilized that allowed for the development of a product that users could trust to work. New algorithms were verified theoretically and experimentally before being added to the source code. Beta test versions of the algorithms were extensively tested by various organizations composed of users from NASA, the DoD, and industry. Bug reporting, tracking, and resolutions were included in the software quality assurance process.

SC3D, which is comprised of approximately 64,000 lines of code, has met the required specifications regarding performance, functionality, and reliability. The initial performance requirement called for refresh rates between five hertz and 10 hertz. Testing indicates that the SC3D system provides much better performance than this, with refresh rates between 10 hertz and 30 hertz. The system functionality requirement called for enhanced SA to be provided to operators of remotely piloted vehicles. The successful integration and use by multiple military branches, various NASA programs, and many other government, commercial, and education organizations is proof that the system can provide windowless cockpit and remote ground station (tele-operation) visualization. The embracement of the SC3D system by many organizations, where lives depend on the reliability of the information, is a testament to its excellence.

Weekly and quarterly reports were used as part of an Earned Value Management process to track schedule, budget, and deliveries. Weekly reports highlighted the technical accomplishment for the week and outlined what would be done the following week. Quarterly reports were comprehensive and included the money allocated for the quarter, percentage of physical work completed to date, and the work that would be performed the upcoming quarter. The average Cost Performance Index and average Schedule Performance Index for the third SC3D spiral development cycle of the project were 0.99 and 1.0 respectively. The SC3D project met or exceeded all scheduled software, hardware, and documentation deliveries.

User Base

A testament to the enormous capabilities, quality, and reliability provided by the SC3D system is the large number and diversity of users that have enjoyed the benefits provided by the SC3D system. SC3D usage consists of individuals from more than 90 different organizations, including the following: DoD, NASA, education institutions, research institutions, European Space Administration, and many industrial organizations.

The SC3D system supports several branches of the military by providing functionality in many major areas that include combat/reconnaissance visualization for unmanned aerial vehicle (UAV) operators, development of new human interface concepts for pilots, training of new and experienced pilots, and supporting troops in combat. SC3D will also play an important role as a recruitment tool such as the following:

- The Advanced Systems and Concepts Office, in the Assistant Deputy Undersecretary of Defense (ADUSD) office, has been investigating many uses of SC3D for the warfighter. Cmdr. Thomas Moore, special assistant in the ADUSD office noted:

  > SmartCam3D is truly a one-of-a-kind breakthrough technology in situation awareness. There is no other software currently available that provides these benefits to the warfighter. [1]

- The Air Force Research Lab (AFRL) is using SC3D technology to develop new interface paradigms for UAV operators. The AFRL Interactive Visual Interface Environment for UAVs program has integrated the SC3D system with a Predator simulator to create an excellent environment to test new human interface concepts related to the tele-operation of Predator UAVs. As Dr. Mark Draper (Human Effectiveness Directorate - AFRL) noted:

  > Within the past year, this effort has been successfully demonstrated and briefed to the commanders of AFMC, AFRL, ASC/RA and ACC/DR UAV SMO among others … The Air Force clearly sees the potential that SmartCam3D technology provides. [2]

- The SC3D system has provided commanders/soldiers with enhanced battle SA. Operators with the 1-14 Cavalry in Iraq noted:

  > Our current mission is Force/Perimeter Protection since our immediate threat is very real and very close. Every OP [Observation Post] that hears or sees something we respond to. Having the SmartCam-3D overlays assisted in our response time. [3]

- The SC3D system will become an extremely important recruiting tool. As a project office representative with the 1st Stryker Brigade Combat Team noted:

  > SmartCam3D is a big hit with the soldiers. SmartCam3D could become the Army's biggest recruiting tool to encourage young video gamers to enlist and re-enlist. [4]

Another major user of SC3D is NASA, which is using the system to support endeavors related to flight/aviation safety, visualization requirements definition for future spacecraft, mission planning, flight safety/operations, and visualization for future Mars and Lunar rovers. SC3D technology, which includes VisualFlight and Landform, is also supporting various endeavors related to basic research, accident investigations, educational outreach, entertainment, and other efforts.

Summary

SC3D is the first real-time HSVS system. Its component-based architecture allows it to be easily incorporated into other software applications. The processes, technologies, and early inclusion of user feedback played a critical role in the highly successful nature of the SC3D system. Its numerous innovations make it easy to use and provide numerous capabilities not offered in any other visualization system currently available. SC3D technology reduces uncertainties, minimizes costs, and improves safety.

Its large user base includes many federal government agencies, various education institutions, the European Space Administration, many commercial companies, and the entertainment industry. This large user base and highly successful flight utilization are a testament to the usability and quality of the software. The improved SA that SC3D provides has proven very valuable to U.S. troops in combat operations, reconnaissance missions, and is helping to neutralize terrorist threats. As a project office representative with the 1st Stryker Brigade Combat Team noted, "This software [SC3D] will save lives."◆

References

1. Moore, Thomas. Letter of recommendation for Software of the Year Competition. Apr. 2004.
2. Draper, Mark. Letter of recommendation. 24 Mar. 2004.
3. Geisler, David. "Re: Operators with the 1-14 Calvary in Iraq." E-mail to J. Allison, M. Righter, R. Smith, T. Schwierling, and T. Ostheller. 22 Dec. 2003.
4. Black, Brian. Letter of recommendation. Mar. 2004.

Project Points of Contact

Frank Delgado
NASA
MC/ER2
2101 Nasa PKWY
Houston, TX 77058
Phone: (281) 483-9077
Fax: (281) 483-3204
E-mail: francisco.j.delgado@nasa.gov

Mike Abernathy
Rapid Imaging Software
1318 Ridgecrest PL SE
Albuquerque, NM 87108
Phone: (505) 265-7020
Fax: (505) 265-7054
E-mail: mikea@landform.com

Janis White
Statistical Consulting
2726 Monterey SE
Albuquerque, NM 87108
Phone: (505) 268-5124
E-mail: janis@landform.com

WARSIM Enters the Scene in Army Training

Ed Payne
Lockheed Martin Simulation, Training, and Support

Col. Kevin Dietrick
Program Executive Office for Simulation, Training, and Instrumentation

The Warfighter's Simulation (WARSIM) is the U.S. Army's next generation constructive simulation capability. Revolutionary in scope, WARSIM allows commanders and their staffs to train in the contemporary environment they will face in Iraq, Afghanistan, and other trouble spots in the world. WARSIM provides additional needed realism, will decrease the resources' need to train, and when combined with the other systems of the Army's Constructive Training Federation will provide all the capabilities required for a fully integrated live-virtual-constructive training environment.

The Warfighter's Simulation (WARSIM) is a computer-based simulation that will revolutionize the way the commander and staff train and conduct mission rehearsal in the contemporary operating environment. The program mission is to design, develop, produce, and deploy a training simulation system to support U.S. Army commanders and their staffs. The WARSIM architecture provides flexibility to interface with other live, virtual, and constructive training simulations and to simultaneously interoperate with the organic Command, Control, Communications, Computer, and Intelligence systems and equipment of the training audience. WARSIM, a software-intensive system with more than two million lines of code, provides the Army's next generation command-and-control training environment.

Recent studies recognize a military training revolution that occurred when the Combat Training Centers (CTCs) were created in the 1980s [1]. The collective training at the CTC gives U.S. forces a decided advantage over its adversaries. Those same studies cite the need for a second revolution in training, one that will expand the collective training success of the CTC to home station and deployed units.

WARSIM is part of that second training revolution: It provides a realistic battlefield environment that more closely matches the contemporary operating environment encountered today in Iraq and Afghanistan. WARSIM is perfectly suited to train the geographically dispersed modular Army; Brigade combat teams from different geographic areas will be expected to join in forming Army or Joint Forces command elements. This will demand a distributed training and mission rehearsal capability that WARSIM brings, and with the high operational tempo of today's units, WARSIM will reduce overhead personnel requirements typically levied on training units.

The U.S. Army Program Executive Office for Simulation, Training, and Instrumentation (PEO STRI), the WARSIM acquisition agent, will meet the Army's training requirements with WARSIM by linking to Army and joint training simulations in the near term, and simultaneously developing increased simulation capabilities applicable to both Army and joint training. WARSIM has an open architecture that enables it to federate with other existing or future simulations.

WARSIM is presently in the engineering and manufacturing development phase of acquisition. The prime contractor is Lockheed Martin Simulation, Training, and Support (LM-STS), which leads an industry team. The intelligence subcomponent of WARSIM was separately developed to accommodate top-secret/sensitive compartmented information requirements and is now completely integrated within WARSIM. The intelligence subcomponent is an integral part of the overall Army strategy for WARSIM.

The initial version of WARSIM was delivered to Fort Leavenworth, Kan., in December 2004. Several factors contributed to the success of the program and the satisfaction expressed by the WARSIM customer. Key among these are the application of concurrent engineering, co-location of project stakeholders, creation of an environment where subcontractors are teamed with the prime contractor, dedication to quantitative management of program processes and measures at Capability Maturity Model® Integration Level 4, an innovative approach to integration and testing, and careful management of the program cost and schedule.

Concurrent Engineering

From its inception, WARSIM has been dedicated to the concept of concurrent engineering where engineering, program management, acquisition, and user representatives interact throughout the development – from requirements analysis through integration and test – as members of integrated development teams (IDTs). The entire WARSIM program is organized using IDTs. Every WARSIM IDT includes members from PEO STRI – the materiel developer – the National Simulation Center, the combat developer, and the Lockheed Martin development team. This approach reduces program risk significantly and ensures that every stage of development is validated and correct across the project's stakeholders.

Co-Location

PEO STRI has been co-located with WARSIM developers from the very beginning of the program. Contractor engineers, customers, user representatives, subject matter experts, and program management shared office space at the development site, which allowed stakeholders to attend all meetings, working groups, and ad-hoc sessions; this made all parties truly integrated members of the WARSIM team.

Subcontractor Teammates

A strong industry team developed the WARSIM product, and includes Lockheed Martin (prime), Science Applications International Corporation, Dynamics Research Corporation, Northrop Grumman, Veridian (General Dynamics), and Virtual Technology Corporation. Each contractor brings a high degree of technical expertise and competence to the program. Another key to the successful teaming approach is that all subcontractors are co-located at the Lockheed Martin facility, sharing offices with Lockheed Martin engineers. Responsibility for development of work products is shared across the teams, with members from each company developing each computer software configuration item. All teammates followed the same Lockheed Martin standard engineering processes.

Quantitative Management

WARSIM's *project management tracking metrics* were developed by analyzing Lockheed Martin Simulation, Training, and Support organizational goals, customer needs as documented in the contractual requirements, and industry standards. Using these goals, the program established quantitative management metrics to objectively monitor process and product quality. These measurements allowed management to assess WARSIM's program status and identify program risks, which then were managed and mitigated in accordance with the WARSIM risk management process.

Quantitative management allowed both WARSIM management and the WARSIM customer to determine in real time that the processes and product development were meeting the project's goals, or to take corrective action as necessary.

Integration and Test

Eighteen months prior to delivery, the program transitioned from a standard development effort to one focused on evaluating the system from the user's perspective. To accomplish this, an integration and test operability team was established to lead these evaluation efforts with all other engineering functions operating in support.

The *operability team* was tasked with testing the system as it will be used once delivered to the users. This team was composed of skilled individuals with a strong background in Army training that worked closely with the user's representatives at the National Simulation Center (NSC) to develop thread tests of the functionality. A primary focus of the operability testing was to evaluate and enhance system usability. During this period, the operability team set the priority for changes and worked side by side with the software developers to define the desired changes.

Key to the concept of operability testing was the hosting of frequent Continual User Assessments in the development environment. The benefit is win-win for both users and developers: Users' training time is greatly reduced as they learn the system as it is built; developers gain valuable user feedback and validation while developing code for the next WARSIM build.

Another benefit of active user participation was an improved understanding of requirements. With a better understanding of the users' needs, software developers would often prototype a capability and demonstrate it to the user – a process that turned out to be extremely efficient – allowing the software team to add more capability to the final product in less time.

Three months prior to delivery, the WARSIM integration and test team held a three-week culminating integration event, commonly referred to as the September '04 Event (S4E). Its purpose was to test the system in its intended configuration and under exercise-like conditions prior to delievery. This event was conceived as a process improvement to anticipate problems observed in the user's environment not experienced in the integration lab environment.

WARSIM leadership are, from left, Greg Manross, WARSIM project director, project manager Constructive Simulation PEO STRI; Debra Palmer, vice president Business Development and Advanced Programs, LM-STS; Ed Payne, WARSIM program manager, LM-STS; Col. Kevin Dietrick, project manager Constructive Simulation, PEO STRI; Dr. Jim Blake, PEO STRI program executive officer.

To prepare for the S4E, the program built a new, classified facility that mirrored the WARSIM installation at the NSC. Twenty-four experts from the Battle Command Training Program, who were previously unfamiliar with WARSIM, participated in the event to ensure the testing was realistic and valid; the successful outcome of the event enabled WARSIM to identify a final set of changes and quickly incorporate them in the months prior to delivery. Because of its thorough user-oriented focus on integration, the program is confident of continued success as it moves through the next phase of user testing.

Schedule and Cost

WARSIM has always met or exceeded its commitments to cost, schedule, and performance. After termination of the Joint Simulation System (JSIMS) program in 2002, WARSIM's program was re-baselined to provide a complete training system to the Army. The program made major schedule, budget, and staffing adjustments to accept additional requirements from the JSIMS program. The contractor delivered a complete training system while holding to the committed delivery schedule within government funding constraints. WARSIM delivered its software on Dec. 17, 2004 and exceeded the user's expectations.

Conclusion

WARSIM is a key enabling program for the training of the Army's current and future force commanders and staff. It is a critical component in the Army Constructive Training Federation (ACTF) that will help to bring about a second revolution in military training. WARSIM will fulfill the Army requirement for training its forces in all aspects of command and control. ACTF models will provide full training functionality for leader and battle staff computer-based simulation training throughout the Army, joint, interagency, intergovernmental, and multinational spectra. WARSIM's contributions to training today's Army, and tomorrow's future forces, are just starting to be realized.◆

Reference

1. Defense Science Board. "Training for Future Conflicts." Washington: Defense Science Board, June 2003 <www.

Project Point of Contact

Mike Haddad
PEO STRI
12350 Research PKWY
Orlando, FL 32826-3276
Phone: (407) 384-3853
Fax: (407) 384-3660
E-mail: mike.haddad@peostri.army.mil

The Myth of the Best Practices Silver Bullet

Michael W. Evans and Corinne Segura
American Systems Corporation, Inc.

Frank Doherty
Space and Naval Warfare Systems Command

For many years, there has been much work attempting to identify a set of best practices that software-intensive projects could apply to aid in the acquisition, production, or upgrade of software. Spurred on by the 1987 and 2001 Defense Science Board findings, efforts conducted by the Software Engineering Institute and the Software Program Managers Network have identified and documented specific practices that have had apparent success in lowering project risk, improving cost and schedule performance, and enhancing user satisfaction. Since Section 804 of the National Defense Authorization Act for Fiscal Year 2003 was enacted on Dec. 2, 2002 and became law, there has been much activity in this area, particularly in the Department of Defense and its various services. This article explores some of these efforts, looks at the practices that have resulted, and attempts to examine certain key relationships that must be considered when applying them to projects.

Best practices are often looked on as the *Holy Grail* of process improvement, the silver bullet that will cure all ills. A manager might reasonably ask, "After all, couldn't I expect the same degree of success if I use the same processes in the management, engineering, assurance, and tracking of the project?" The answer is an unqualified *maybe*.

Indeed, far from being a silver bullet, there is some evidence that the term *best practices* lacks significant meaning. In 2001, Dr. Richard Turner conducted a study for the Department of Defense (DoD) [1] to identify credible best practices that could improve performance, predictability, quality, and operational effectiveness while lowering risk, shortening schedules, and reducing development costs. As a result of this study, Turner concluded that because the term *best practices* is consistently misused, it is misleading at best and useless at worst. It has become a catchall phrase that bundles diverse ideas about practices and frameworks, and is used by some to legitimize unproven practices, tools, or processes.

Unproven practices, while not designated as best, are often essential components of successful projects. They simply do not have a pedigree outside of the project to which they are being applied.

Practice Relationships

Software managers must not assume that just because a certain practice has been labeled *best* that it will indeed improve the performance, predictability, quality, and operational effectiveness of the software they are responsible for producing. Nor does it mean they should view these practices with outright suspicion, but only that they must understand the advantages and disadvantages of the practices for their particular projects and how they can be usefully adapted to the various needs of their organization.

To understand why so-called best practices are not a silver bullet, it is important to understand the difference between a process and a practice. A process is a set of interrelated resources and activities that transform inputs into outputs. When used in a consistently formal manner, these resources and activities tend to increase quality, shorten schedules, and lower cost and risk. Processes are used to conduct business, and they support a unique organizational culture. Practices are disciplines, methods, tools, or techniques that are used to accomplish a specific function or set of functions in a project environment. A process can include multiple practices.

A process can be considered a plan in that it describes what must be done to obtain an output and provides the framework needed to accomplish the necessary tasks. Practices define the manner in which the tasks must be conducted. Both are critical to a project's success because what is to be done must be planned, and how the plan will be accomplished must be defined. Moreover, practices must be adapted to the organization that uses them, and the relationships between the practices must be understood and managed if the expected benefits are to be realized. It is incumbent upon the manager to choose practices that are appropriate to the level of the organization that will implement them and adapt them as necessary. For example, a practice intended to meet the configuration management needs of the acquisition layer of an organization would not necessarily meet the needs of the development layer, but with a proper understanding of its uses, a manager could adapt the practice to meet the needs of both.

A typical program has several layers, each of which has different requirements and constraints regarding the processes and practices used and their implementation. The top layer, the user organization, requires deployment of a product or service that is responsive to the operational and support requirements of the user community. The needs of the user community form the baseline from which required practices are defined and implemented. The user organization requires practices that (1) capture, characterize, and control the user's operational requirements and constraints; (2) define interoperability and system interface requirements; (3) identify how these requirements are qualified and inserted into the operational environment; and (4) define how these elements are documented, maintained, and updated.

The next layer of the program, the acquisition organization, is chartered to acquire the right product at the lowest cost and in the shortest time to satisfy a specified user requirement. To do so, the acquisition organization works with the user to define what is required. It then converts the user needs into functional, engineering, product assurance and support requirements and constraints, and plans and implements processes – supported by practices – to acquire a system, software product, or services that will satisfy the user needs. The acquisition organization next prepares and issues documentation that establishes agreements between the acquirer and supplier(s), selects a supplier(s), and manages the acquisition process until the product or service is accepted.

The acquisition organization requires practices that facilitate the acquisition of the product or service at the lowest risk. It needs practices to collect and evaluate quantitative indicators of both project performance and product quality; to assure the product's quality based on the quantitative indicators; to monitor developer performance; and to facilitate delivery, acceptance, and deployment of the product or service.

The next layer of the program, the development organization, is chartered to build a product in a manner that is consistent with established agreements and specifications and that maximizes profit and meets all commitments and agreements. The development organization needs practices that (1) specify architectures; (2) define, expand, and control specific engineering requirements that are traceable to those provided by the acquirer; (3) control and manage the development process; (4) monitor the quality of the products being developed or received from suppliers; (5) collect quantitative information from implemented practices to ascertain process effectiveness; (6) monitor cost and schedule performance; and (7) monitor development risk against progress toward established requirements.

Finally, support organizations – such as independent test, logistics, installation, product maintenance, etc. – require practices that enable processes, and provide management visibility into and control over the quality of the services provided and the risks that must be addressed.

Given these various needs, it is easy to see that a single set of best practices would be impossible to implement, and that any set of best practices must be adapted to the organizational layer that implements them. For example, best practices related to configuration management necessarily must be adapted to meet the needs of the user level, the supplier level, and the development level. Nevertheless, configuration management can be considered a necessary best practice that all organizations should address if they expect to succeed.

Once adapted to the needs of the various organizations, the practices must be integrated if the project is to progress effectively both within an organization and between organizations. For example, the requirements management process area may require effective implementation of the requirements definition, configuration management, defect identification and removal, and user involvement practice areas to accomplish the process.

In addition, the many practices a particular organization uses must interact with other practices, which are often unrelated, to provide seamless and effective support to the overall process. For example, if a particular practice identifies defects early in the development process, configuration management should account for this, and the software manager must adapt practices to remove defects accordingly to ensure potential efficiencies are realized.

Sample Practices

Table 1 on page 16 lists several project approaches or best practices that have originated from initiatives conducted by various organizations. This list is a small sample of the hundreds identified in the literature as being best [2]. These practices can be categorized by their typical application or use:

- **Policy Requirement.** A policy requirement defines a basic requirement that all program organizations must meet. This category focuses on a desired outcome and typically does not define specific processes or practices.
- **Organizational Concept.** These are general principles that are used to organize the project, allocate resources and responsibility, enable communications, and effect work assignment.
- **General Strategy.** This is a strategy that applies to all organizational components of a project but must be adapted to the needs of a particular organization to be effective. A general strategy, for example, might be that all projects apply continuous risk management to prevent negative consequences from unanticipated issues. Although related to the risk management practices of other organizations, specific implementation of the general strategy within an organization will depend on the organization's particular charter, culture, commitments, and constraints.
- **Business Strategy.** This is a strategy that defines how to accomplish specific business tasks.
- **Acquisition Framework.** This is a structure the acquisition organization uses to acquire, manage, and control the products or services and ensure they are responsive to user needs. It usually consists of activities, specifications, reviews, and reporting requirements.
- **Acquisition Strategy.** An overall statement of how an organization will acquire products and services consistent with user needs and requirements is an acquisition strategy. Expressed in general terms, it typically describes requirements that will constrain the selection and adaptation of processes and practices and defines the goals that must be met to satisfy validated needs as well as to maximize affordability.
- **General Practice.** This is a practice that supports every organizational level. Specific application of the practice will differ according to the needs of the user, acquirer, supplier, and support layers of the program organization.
- **Development Practice.** This is a practice that predominately supports the supplier's requirements.
- **Acquisition Practice.** This is a practice that ensures an acquisition is conducted effectively by the acquisition organization as it monitors the project and controls the suppliers. The practices are structured to evaluate and receive products and services rather than develop and deliver them.
- **Maturity Model.** This is used to evaluate the process maturity of an organization to determine the potential risk of a process and the potential to use it successfully in other circumstances.

Table 1 identifies project approaches, i.e., best practices that have been extracted from several sources, including the Software Program Managers Network 16 Point Plan, the DoD Best Practices Study conducted by Dr. Richard Turner, the European Software Institute 1977 Software Best Practice Questionnaire Analysis of Results, and various other studies. Some of the approaches are related to policy, others are related to process, and still others are related to strategy. However, they all are important considerations with significant benefits that an organization could use to establish an effective project environment and conduct the activities identified in their project plan.

As Norm Brown posits in *IEEE Software* [3], the definition of a small number of relevant best practices can have a significant effect on the success of a project, but only if the practices are tailored to the needs and culture of the organization that will use them. In Table 1, we have identified the practice; the source, including the primary reference that was used to identify it; and the general classification of the practice. Turner's dissertation [4], "Implementation of Best Practices in U.S. Department of Defense Software-Intensive Systems Acquisitions," is identified as the source for many of the practices included in the table. For readers' convenience, we have included the source reference for the specific practices identified in the Turner dissertation.

Basic Considerations for Implementing Practices

The Turner study discusses what must be

Practice Identification	Practice Source	Practice Category
Establish Clear Goals and Decision Points	(DSBTF, 5000.2R) [5,6]	Policy Requirement
Treat People as the Most Important Resource	SPMN 16 Point Plan [7]	Policy Requirement
Common Management and Manufacturing Systems	(5000.2R) [6]	Policy Requirement
Integrated Product and Process Development	(5000.2R, Reifer) [6,4]	Organizational Concept
Appointing Project Managers for Each Project	ESI [8]	Organizational Concept
Software Quality Assurance Function with Independent Reporting Line	ESI [8]	Organizational Concept
Training New Project Managers	ESI [8]	Organizational Concept
Have a Formal Review or Handover of Deliverables From One Project Group to Another	ESI [8]	Organizational Concept
Maintaining Awareness of New Development Technologies	ESI [8]	General Strategy
Ensuring User Input at All Stages of the Project	ESI [8]	General Strategy
Assess Viability, Risks, and Benefits Before Committing to a Project	ESI [8]	General Strategy
Conduct Periodic Reviews of the Status of Projects	ESI [8]	General Strategy
Demonstration-Based Reviews (Including Executable Architectures)	(Royce, DSBTF, ISO) [4,5,9]	General Strategy
Conduct Inspection and Walkthroughs at Each Stage	ESI [8]	General Strategy
Require Structured Development Methods (Iterative Processes)	(Royce, DSBTF) [4,5]	General Strategy
Plan for Technology Insertion	(5000.2R, DSMC) [6,10]	General Strategy
Commercial and Specifications and Standards/Open Systems	(5000.2R, Anderson, Jones) [6,4]	General Strategy
Statistical Process Control	Reifer [4]	General Strategy
Capture Artifacts in Rigorous, Model-Based Notation	(Royce, DSBTF) [4,5]	General Strategy
Strategic Partnering	Reifer [4]	Business Strategy
Relationship Management	Reifer [4]	Business Strategy
Market Watch	Reifer [4]	Business Strategy
Enterprise-Wide Licensing	Reifer [4]	Business Strategy
Independent Expert Reviews/SCEs	(5000.2R, DSBTF, DSMC, Jones) [6,5,10,4]	Acquisition Framework
Performance-Based Specifications	(5000.2R, Anderson, Reifer) [6, 4]	Acquisition Framework
Use Past Performance	(5000.2R, DSBTF, Anderson, Reifer) [6,5,4]	Acquisition Framework
Leverage Commercial Off-the-Shelf (COTS Items, Non-Developmental Items (NDI)	(Anderson, Reifer) [4]	Acquisition Framework
Ensure that Subcontractors Follow Formal Processes	ESI [8]	Acquisition Strategy
Best Value Awards	(5000.2R, DSBTF, Anderson, Reifer) [6,5,4]	Acquisition Strategy
Adopt Continuous Risk Management	SPMN 16 Point Plan [7]	General Practice
Estimate Empirically Cost and Schedule	SPMN 16 Point Plan [7]	General Practice
Use Metrics to Manage	SPMN 16 Point Plan [7]	General Practice
Track Earned Value	SPMN 16 Point Plan [7]	General Practice
Track Defects Against Quality Targets	SPMN 16 Point Plan [7]	General Practice
Adopt Life Cycle Configuration Management	SPMN 16 Point Plan [7]	General Practice
Manage and Trace Requirements	SPMN 16 Point Plan [7]	General Practice
Ensure Data and Database Operability	SPMN 16 Point Plan [7]	General Practice
Assess Reuse Risks and Costs	SPMN 16 Point Plan [7]	General Practice
Inspect Requirements and Design	SPMN 16 Point Plan [7]	General Practice
Requirements Trade-Off/Negotiation	(DSBTF, Royce) [5,4]	General Practice
Perform Independent Testing	ESI [8]	General Practice
Have Formal Methods of Estimating Software Size	ESI [8]	General Practice
Formally Review the Functionality of the System the Software Replaces	ESI [8]	General Practice
Use Formal Methods to Estimate Schedule and Cost	ESI [8]	General Practice
Use System-Based Software Design	SPMN 16 Point Plan [7]	Development Practice
Define and Control Interfaces	SPMN 16 Point Plan [7]	Development Practice
Design Twice, Code Once	SPMN 16 Point Plan [7]	Development Practice
Manage Testing as a Continuous Process	SPMN 16 Point Plan [7]	Development Practice
Compile and Smoke Test Frequently	SPMN 16 Point Plan [7]	Development Practice
Architecture-First Approach	(Royce, DSMC, DSBF) [4,10, 5]	Development Practice
Have Common Coding Standards for Projects	ESI [8]	Development Practice
Plan Testing Before Coding	ESI [8]	Development Practice
Establish Reliability and Stress Margins	Doherty SEI [11]	Acquisition Practice
Personal Software Process^{SM}/Team Software Process^{SM} Practices	Humphrey [12,13]	Maturity Model
Acquisition Process Improvement	SEI [14]	Maturity Model
Contractor Capability Evaluation	SEI [15]	Maturity Model

DSBTF: Defense Science Board Task Force
ESI: Enterprise Software Initiative
SCE: Software Capability Evaluation
DSMC: Defense Systems Management College

^{SM} Team Software Process and Personal Software Process are service marks of Carnegie Mellon University.

Table 1: *Examples of Best Practices*

considered to successfully migrate practices from project to project. Before management selects specific practices to accomplish a project, it must address the organization's culture, attitude, and experience, and most certainly these factors must be considered when management devises its strategy to introduce the practices selected [16].

For example, early identification and removal of defects through consistent application of structured inspections is a valuable goal. However, if the organization's management presumes that its engineers will not make mistakes, and if it will not adequately fund this practice consistently from concept through delivery, the practice is not realistic given the organization's culture.

Not only must management understand its organization's culture, it must also understand the true costs and risks associated with implementing a practice before it commits to using the practice. That is, management must honestly assess and understand the following:

- The effect of the practice on project teams regarding their possible resistance and the potential for increased productivity.
- The costs associated with the practice and the potential return on investment.
- The cost required to train those who will apply the practice.
- The availability and cost of associated tools.
- Potential barriers to implement the practice and its application.
- The validity and general acceptance of the practice within the industry.
- The effect of the practice on related and interfacing practices, processes, and tools.
- The degree of management and staff commitment to the practice, and what factors led them to commit to the practice.

It is critical that management understand the true costs and impacts of the practices it implements, whether they have been proven in other environments or not. If management implements a practice without understanding its costs and effects, it could well be incompletely or haphazardly implemented, and the project will suffer as a result. Indeed, if the implementation is incomplete, poorly planned, or otherwise improper, or, worse, if the practice must be replaced mid-project, the effects – such as poor staff morale and productivity, tool replacement, retraining, and file or artifact conversion – can be devastating.

In addition to these considerations, projects with substantial software content that show evidence of certain characteristics are poor candidates for the reasoned application of best practices. These characteristics, which were documented in the April 2002 CrossTalk [17], are the following:

- Unwarranted optimism and unrealistic executive management expectations.
- Late decision-making.
- Inappropriate use of the standard software process.
- Missing or inadequately implemented program activities.
- Lack of leadership.
- Early declarations of victory.
- An absence of risk management, which could convince managers and staff they can accomplish unrealistic objectives given the actual project circumstances.

These underlying attitudes, which can be understood to presume project success and minimal risk, often convince project management and staff that they need not adequately plan to implement a practice and develop process standards. Such complacency can be costly.

Given the fact that a best practices silver bullet does not exist, organizations cannot unthinkingly adopt a pro forma approach to project completion and assume the practices they implement will automatically succeed. To truly succeed, management must understand how the practices they use will work within their unique organization, which will lead to a solid project management foundation and will in turn positively affect the bottom line regarding productivity, quality, timeliness, and user satisfaction.◆

References

1. Turner, Ph.D., Richard. "A Study of Best Practice Adoption By Defense Acquisition Programs." CrossTalk May 2002: 4-8 <www.stsc.hill.af.mil/crosstalk/2002/05/turner.html>.
2. Brotbeck, George, Tom Miller, and Dr. Joyce Statz. "A Survey of Current Best Practices and Utilization of Standards in the Public and Private Sectors." TeraQuest Metrics, Inc., Nov., 1999.
3. Brown, Norm. "Industrial-Strength Management Strategies." IEEE Software July 1996.
4. Turner, Ph.D., Richard. "Implementation of Best Practices in U.S. Department of Defense Software-Intensive Systems Acquisitions." Washington: George Washington University, 31 Jan. 2002 <www.gold practices.com/survey/ turner/turner_dissertation.pdf>.
5. Defense Science Board. "Report of the Defense Science Board Task Force on Defense Software." Washington: Department of Defense, 2000.
6. Department of Defense. Interim Defense Acquisition Guidebook (Formerly DoD 5000.2-R). Washington: DoD, 30 Oct. 2002 <www.acq.osd.mil/dpap/Docs/InterimGuidebook.doc>.
7. Software Program Managers Network. 16 Critical Software Practices For Implementing Performance-Based Management. Vers. 3.0., Arlington, VA: Integrated Computer Engineering, Inc., 2 Aug. 2000.
8. European Software Institute. "1997 Software Best Practice Questionnaire." ESI-1997 Code Version. Bizkaia, Spain. ESI, Dec. 1997.
9. International Organization for Standardization. "Information Technology: Software Life-Cycle Processes." Geneva, Switzerland: ISO, and International Electrotechnical Commission, 1995.
10. Defense Systems Management College. "Best Practices and Course Summary." Software Acquisition Management 101. Ft. Belvoir, VA: Defense Systems Management College, 2001.
11. Doherty, Frank, and Corinne C. Segura. "Software Development Challenges Within the DoD Acquisition Framework." Systems and Software Technology Conference, Salt Lake City, UT. May 2004.
12. Humphrey, Watts. Introduction to the Personal Software Process^{SM}. Addison-Wesley 1997.
13. Humphrey, Watts. Introduction to the Team Software Process^{SM}. Addison-Wesley 2000.
14. Bernard, Tom, Brian Gallagher, Roger Bate, and Hal Wilson. "CMMI Acquisition Model, Vers. 1.0." CMU/SEI-2004-TR-001. Pittsburgh, PA: Software Engineering Institute, 2004 <www.sei.cmu.edu/publications/documents/04.reports/04tr001.html>.
15. Software Engineering Institute. "Word Versions of CMMI® Models." Pittsburgh, PA: SEI <www.sei.cmu.edu/cmmi/models/model-components-word.html>.
16. Evans, Michael. "Managing Acquisition Risk Through Application of Proven Best Practices." Systems and Software Technology Conference, Salt Lake City, UT. Apr., 2004.
17. Evans, Michael, Alex Abela, and Tom Beltz. "Seven Characteristics of Dysfunctional Software Projects." CrossTalk Apr. 2002 <www.stsc.hill.af.mil/crosstalk/2002/04/evans. html>.

About the Authors

Michael W. Evans is a Senior Vice President at ASC Inc. Previously he was president of Integrated Computer Engineering, Inc. He is experienced in providing direct technical services and support in software engineering methods and processes, software standards, quality assurance, configuration management, and testing. He is co-founder and prime contractor for the Software Program Managers Network, the driving force behind the Department of Defense's Software Acquisition Best Practices Initiative. He is the author of "Principles of Productive Software Management," "Productive Test Management," "Software Quality Assurance and Management," and "The Software Factory."

America Systems Corporation
88674 Shoreline Loop
Florence, OR 97439
E-mail: candca@aol.com

Corinne Segura is a Project Manager for American Systems Corporation (ASC). For the past four years, she has provided technical and managerial support in the areas of process improvement, risk management, program assessments, and test and evaluation. Prior to joining ASC, she served in the United States Navy for 20 years as a Fleet Support Officer. She has a Bachelor of Science in Biology from Northern Illinois University, a Master of Science in Systems Management from the University of Southern California and a Master of Science in Electrical Engineering from the Naval Postgraduate School.

American Systems Corporation
3033 5th AVE STE 205
San Diego, CA 92103
Phone: (619) 208-7140
E-mail: corinne.segura@2asc.com

Frank Doherty is the principal deputy for Transformation at the Space and Naval Warfare Systems Command (SPAWAR), San Diego, Calif. He has been a deputy program manager for Combat Surveillance Systems in the Program Executive Office, Command, Control, Communications, Computers, and Intelligence and Space. With 38 years of government service, Doherty has been lead for Acquisition Streamlining and chief of Industrial Quality and Productivity Division in the Office of the Secretary of Defense; chief, Contract Pricing and Financial Services Division; and director E-3A Procurement.

SPAWAR OOA-2
4301 Pacific HWY
San Diego, CA 92110-3127
Phone: (619) 524-7348
Fax: (619) 524-7374
E-mail: francis.doherty@navy.mil

Measure Like a Fighter Pilot

Joe H. Lindley
Raytheon

The late Col. John Boyd, U.S. Air Force, developed the Observe, Orient, Decide, and Act (OODA) Loop as a strategy for air combat and warfare. It is an elegant, fast decision-loop strategy that is also useful in engaging the measurement and analysis process with the corrective action loops of a development project. The OODA Loop will be introduced in this article along with practical suggestions and lessons learned related to implementation of the measurement and analysis process. These recommendations are based upon years of measurement and analysis experience, recently with a Capability Maturity Model® Integration Level 3 Best Practice measurement and analysis process.

A measurement analyst can face daunting challenges when implementing a measurement and analysis process. Project dynamics and the array of forces that oppose change can easily stall implementation, even when the process is based upon respected guidelines such as Capability Maturity Model® Integration (CMMI®). To be successful, a measurement analyst must have an effective game plan (a focused strategy of when to do what, where).

After years of experience with measurement and analysis, most recently developing a CMMI Level 3 Best Practice measurement program for a large development and integration program, I have accumulated a number of suggestions and lessons learned. I did not have a cogent game plan that I could share with others, however, until I discovered the Observe, Orient, Decide, and Act (OODA) Loop, which explained almost all the successes and failures I have experienced.

The steps of this decision-loop concept provide a forward-leaning framework for action that will bring measurement and analysis to its full potential. The OODA Loop will be used in this article as a framework to relate my first-hand suggestions and lessons learned.

The OODA Loop

Col. John R. Boyd, an accomplished U.S. Air Force fighter pilot and military thinker, originally developed the OODA Loop as a winning strategy for air combat. Later, it was a key part of Boyd's maneuver warfare strategy, which has been embraced at some level by most U.S. armed services. The phrase *getting inside the enemy's decision loop* can be heard in military briefings in recent years – a reference to the OODA Loop. Col. Boyd passed away in 1997 at the age of 70.

OODA Loop Definition

The OODA Loop is a strategy for effective decision making. Figure 1 is a simplified view of the concept. Boyd broke the decision-making process into four steps: Observe, Orient, Decide, and Act, as defined below:

1. **Observe.** Collect and organize information related to recent events, feedback from prior actions, and changes in the environment.
2. **Orient.** Orient to the unfolding environment and current observations. This is the most critical step and is prone to failure. The speed with which orientation is completed is dependent upon the current paradigm (i.e., expectation of what the observations and current environment should be). In simplistic terms, the Orient step will complete in one of three ways. Each is characterized by the speed with which orientation will complete:
   - **Fast.** Observations and environment match expectations (paradigm): Quick, gut-feel, decision/action can be initiated.
   - **Moderate.** Observations and environment differ in some ways from the paradigm: Analysis will be required to develop alternate plans of action and the paradigm may need to be adjusted.
   - **Slow.** Observations and environment differ substantially from the paradigm: The current paradigm may have to be discarded and a new one synthesized.
3. **Decide.** From available actions, select which one to take.
4. **Act.** Take action.

Boyd also asserted the following:

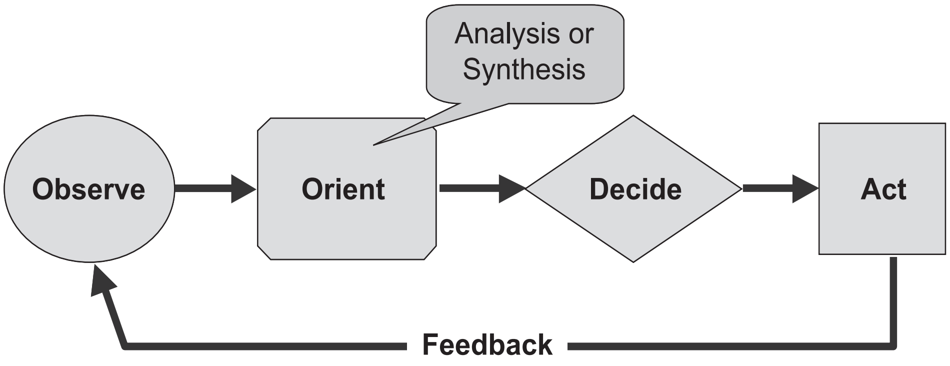
- All human behavior, individual or organizational, can be depicted as a continual cycling through the four steps.
- Success or failure depends upon the relationship of one's own loop to that of an opposing loop.
- Success or failure depends on one's ability to swiftly and accurately complete cycles of one's own loop faster than opposing forces can cycle through their loop.

Jet Fighter Example

The utility of the OODA Loop is best illustrated by an example from Boyd's combat experience in the Korean War – a dogfight between a U.S. F-86E Sabre Jet and a Russian MiG-15. Boyd was interested in a paradox related to these fighters. Sabre Jets were shooting down MiGs at a 10-1 ratio in the Korean War, in spite of the fact that MiGs could out-run and out-turn Sabre Jets. The Sabre Jet provided better visibility and was more responsive than the MiG, but this was not enough to explain its success ratio until the OODA Loop concept demonstrated the power of a fast decision loop.

Each Sabre Jet maneuver is creating a

Figure 1: *The OODA Loop*



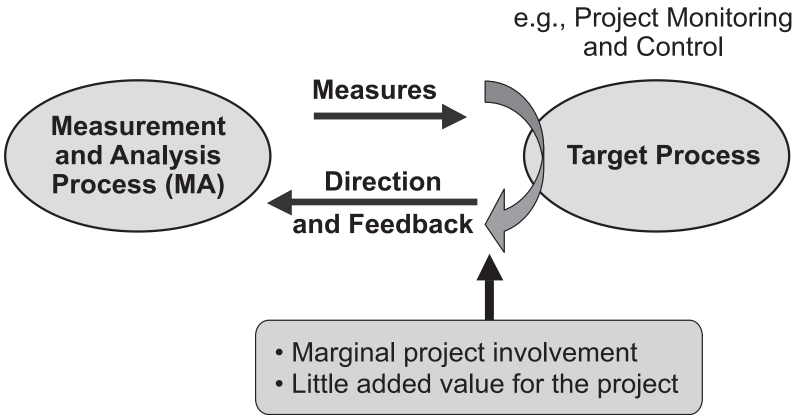

e.g., Project Monitoring and Control
Measures
Measurement and Analysis Process (MA)
Target Process
Direction and Feedback
• Marginal project involvement
• Little added value for the project

Figure 2: *Shallow Measurement Loops Fail*

change in the tactical situation. If the MiG pilot is completing maneuvers (and OODA Loop cycles) at the same pace, he will be able to observe and reorient to each maneuver as it occurs. New observations will generally match his expectations since he is reorienting to each new maneuver. His performance should therefore be on par with the Sabre Jet pilot.

A trained Sabre Jet pilot, however, will use his jet's responsiveness to complete maneuvers faster than the MiG. The MiG pilot, now unable to cycle through the loop as fast as the Sabre Jet, will fail to keep his orientation updated, and will begin to notice a mismatch between what he expects to observe and what he actually observes. Resolving the ambiguity of the mismatch will slow down the MiG pilot's orientation forcing him to cycle through the loop even slower, leading to larger mismatches. For the MiG pilot, time will appear to be compressed, giving him little time to think. For the Sabre Jet pilot, time will seem to almost stretch out and slow down. The Sabre pilot will be able to harass and eventually entrap the MiG pilot to win the engagement.

The Sabre Jet pilot can win by using fast transient maneuvers to degrade the MiG pilot's orientation. For air combat and warfare, speeding up one's own OODA Loop (operating within the enemy's OODA Loop) can be used to literally destroy an enemy's decision-making capability.

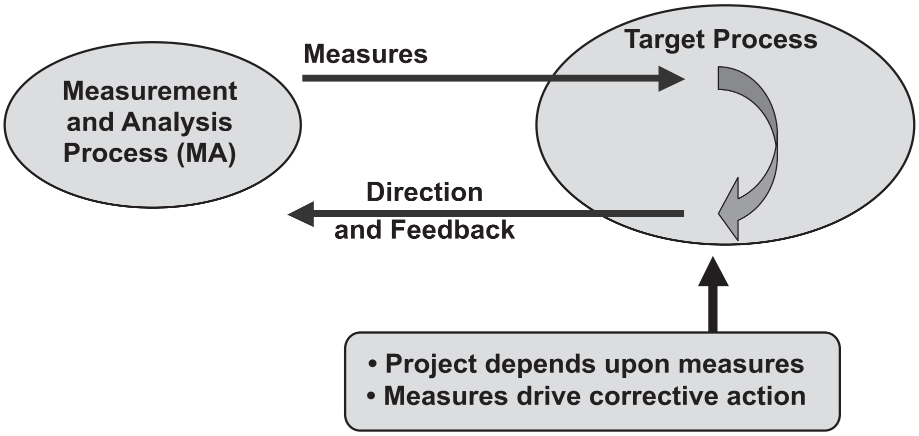
On the other hand, the same idea can be used constructively for engineering process. By operating within the OODA Loop of a target process, a process change leader can make supportive actions to enhance and positively influence the target process decision loop.

The OODA Loop as an Enabler for Measurement and Analysis

Current development processes such as CMMI and Practical Software and Systems Measurement provide excellent guidance on how to plan and conduct measurement and analysis. A critical challenge in implementing these models, however, is interacting with other project processes; in other words, engaging with the managers and other stakeholders who actually initiate corrective action – the end goal for measurement and analysis.

We have all seen projects that *have metrics* but practice a shallow implementation, where metrics are collected and reported but do not have much of an impact on the stakeholders or drive corrective action at all. Figure 2 illustrates this situation. Stakeholders in this environment generally fail to appreciate the value of measurement.

Figure 3: *Deep Measurement Loops Engage and Succeed*


Measures
Target Process
Measurement and Analysis Process (MA)
Direction and Feedback
• Project depends upon measures
• Measures drive corrective action

On the contrary, the measurement loop should extend deep into the target process. Figure 3 illustrates the ideal situation. With a deep measurement loop, the target processes will depend upon measures, and the measures will regularly drive or shape change. In this situation, the measures truly engage target processes. Target process stakeholders will request more measurements over time, raising the importance of measurement in their project activities. At this level of engagement with its target process, measurement and analysis is exciting, vital work.

Driving a measurement loop deep into the target process requires a focused strategy. My suggestion is to engage the target process by getting inside its decision loop. Visualize the decision loop that stakeholders use for their decisions (Observe, Orient, Decide, and Act) and include measurement collection and analysis in the Observe and Orient steps (see Figure 4). Consider this the playing field for the measurement analyst. All the measures provided should be weighed against how they will play in the loop. The measurement analyst should drive his/her own decision loop fast enough to stay *inside* the stakeholders' decision loop to deliver what they need when they need it. The measurement analyst should be making fast transient moves, like a Sabre Jet pilot, to enhance the stakeholder decision loop.

Getting inside the stakeholders' decision loop may be a significant paradigm shift for some measurement analysts. It will pull them away from the relative safety of the numbers and into the project where they will have to find out how to really drive and shape change. Measurement will become more challenging, but much more effective.

Practical Suggestion

A sign that you are truly inside your stakeholders' decision loop is the interaction with the stakeholders. They will ask questions, request and even argue over refinements, and take a real interest in your measures. During my measurement briefings to our customer, my project managers often add commentary of their own. They have played a significant role in refining the measures and are very familiar with my charts.

The OODA Loop Applied to Measurement and Analysis

The OODA Loop is used in measurement

and analysis as a framework for action. Illustrated in Figure 5, the OODA Loop cycle includes the measurement and analysis functions in the observe phase with the remainder of the loop's actions (Orient, Decide, and Act) primarily in the target process. The Orient step is a joint effort, wherein the measurement analyst orients the observations to the stakeholders' perspectives, and the stakeholders then familiarize themselves to the unfolding project environment.

Note that the figure represents a measurement and analysis loop for a single project activity. For a project, many of these will exist. The diagram includes three corollaries associated with the OODA Loop concept, which will be defined in the Corollaries section: tempo, harmony, and ground truth.

Each of the OODA Loop steps (Observe, Orient, Decide, and Act) is defined below, along with related practical suggestions and lessons learned.

Observe

The observe step for the measurement analyst entails the collection and analysis of measurement data.

Practical Suggestion

One of the most difficult and valuable services a measurement analyst can provide is forecasting (e.g., forecasting the expected number of defects that will be encountered during formal test). It is difficult because forecasting requires creative analysis and data collection from a variety of sources. Forecasting is especially valuable because it provides a basis for objectively assessing status, finding leading indicators, and confirming management assumptions.

Orient

Process stakeholders perform the Orient step by orienting to the ongoing situation in preparation for the last steps of the loop, Decide and Act. The measurement analyst plays a key role in the step by orienting the measures for the stakeholders (i.e., making the data meaningful in a perspective that is both familiar and effective in decision making).

Figure 5 includes an extra paradigm loop for the Orient step related to the stakeholder paradigm. This is an important concept. Based on experience and attitude, each stakeholder will have an expectation of how measurement should work, and what the measurements will be. This is the stakeholder paradigm. It will have an impact on how the stakeholder reacts to measures and can either help or hinder measurement. The measurement analyst will generally need to structure measures so that they are aligned with the stakeholder paradigm and, occasionally, work on modifying the stakeholder paradigm.

Practical Suggestion

Provide status data for the whole organization plus drill-downs (filtering or other methods of providing status for lower-level organizations). See Figure 6 for an example. This format is used to brief code-size status for the project (top left chart in Figure 6) and each of the lower-level subsystems. In this way, problems at the subsystem level (such as shown in the bottom right chart of Figure 6) are not missed. The charts are simple and provide limits/plans so that reviewers can objectively assess status.

Lesson Learned

I once failed to ensure that project managers reviewed measurement data at the normal periodic rate. The managers were busy and I did not see anything to worry about, so I did not insist on the normal measures reviews. I realized later that we had missed an opportunity to take corrective action on a problem the managers would have spotted if they had seen the data. I had failed to recognize that it was their orientation, not mine, that was important.

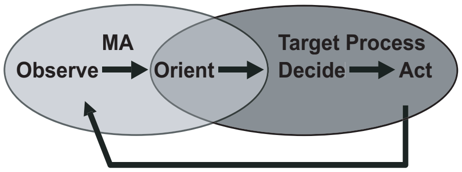


Figure 4: *Engage by Getting Inside the Target OODA Loop*

Decide

Project stakeholders perform this step. The challenge for the measurement analyst is to provide measures that will support reliable and expeditious decision making by the stakeholders.

Practical Suggestion

To optimize decision making, I follow what I call the *Bruno the Trained Ape* rule for

Figure 5: *The OODA Loop for Measurement and Analysis*

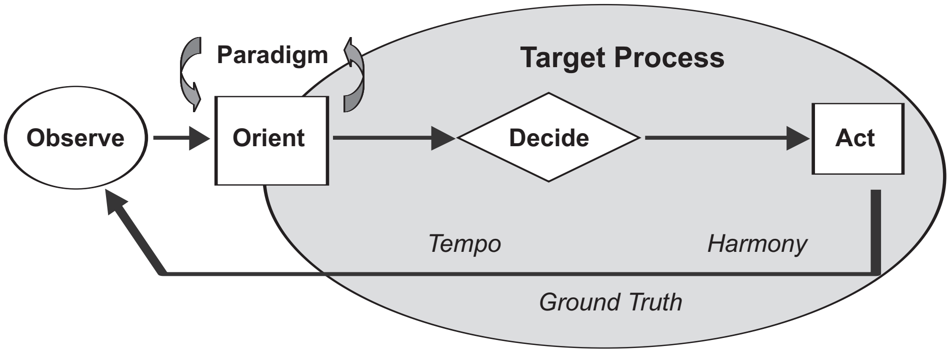
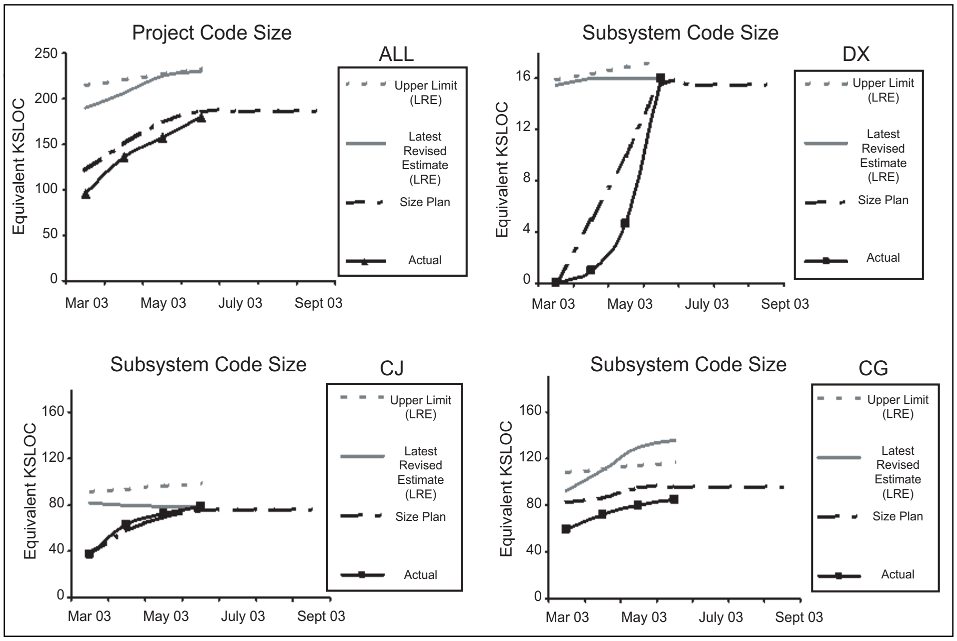


Figure 6: *Example Code Size Charts*



charts. Charts must be so simple that an ape can understand them. This is an exaggeration of course, and has nothing to do with the intelligence of stakeholders. The issue is that busy stakeholders only have a few seconds to assess a chart. The answer needs to jump out at them. Charts should be stripped of anything that is not directly related to the decision at hand.

Lesson Learned

In terms of chart design and its importance in decision making, I have learned a great deal from the books by Edward Tufte [1, 2]. He provides profound suggestions and some interesting examples of lessons learned related to the importance of trimming charts down to only meaningful data.

Act

As with the Decide step, this step is taken by project stakeholders, so the challenge for the measurement analyst is to provide actionable measures that support expeditious corrective action. An analysis that leads to a decision to act is often useless unless a suitable corrective action is identified immediately or shortly thereafter.

Practical Suggestion

I publish integration defect spreadsheets every week to assist stakeholders in assessing the status of defect resolution. A well-known part of these spreadsheets is what the leads call, in jest, the flogging list. It is a listing of defects with a subsystem filter, which will display the details for all defects belonging to a selected subsystem. At weekly meetings, subsystem leads generally report status using the flogging list so that the project manager has the data needed to allocate resources and take immediate corrective action. The flogging list is a good example of an actionable measure.

Corollaries

The framework for action established by the OODA Loop lends itself to the development of corollaries – valuable principles related to the OODA Loop that I have used successfully. Three corollaries are defined below: tempo, harmony, and ground truth. Figure 5 depicts the OODA Loop and the context of these corollaries.

Tempo

Tempo is the goal that OODA Loop timing must match or exceed the timing of the target process. The OODA Loop technique will fail totally unless tempo is adequate. This includes both the measurement cycle time (how often the loop is executed) and the loop cycle time (how quickly each loop cycle is completed).

Practical Suggestion

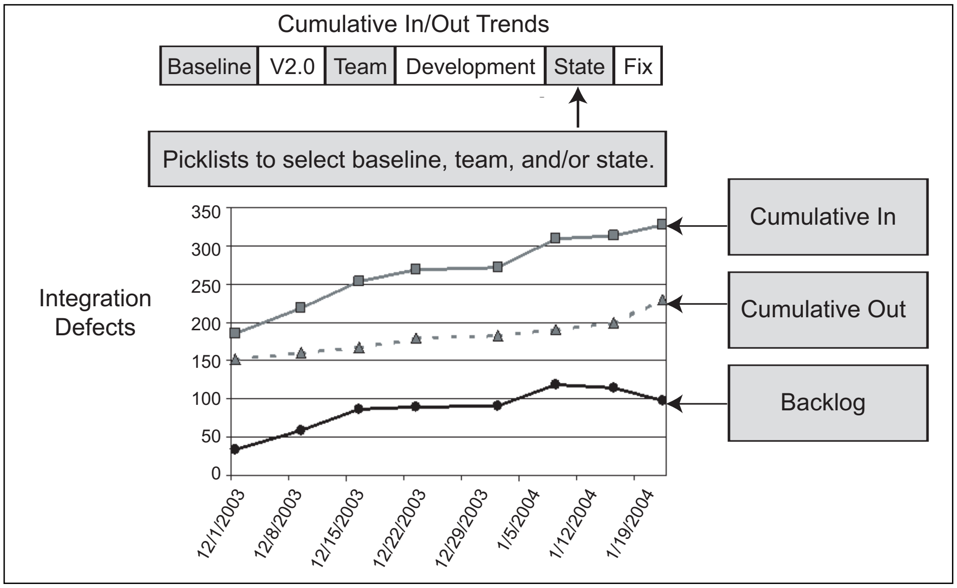
The measurement and analysis OODA Loop must operate within the natural timing of the process being measured:

- Be aware that loop cycle times may be surprisingly short – even with long measurement cycles.
- Develop tools (e.g., macros, scripts, queries) to gain efficiency and reduce the time it takes to respond to ad-hoc management requests.
- Take advanced courses on spreadsheets, databases, etc. to gain proficiency in tool development.

Lesson Learned

I learned the value of tempo the hard way. By the time I completed a full analysis of our inspection process for our first increment, we were halfway through coding on the next increment – too late to help with that increment.

Figure 7: *Example Defect Chart*



Harmony

Harmony is the goal of maintaining a good relationship between the measurement analyst and the project personnel that provide measurement data. Without harmony, data collection will be difficult, and corrective actions may not be constructive.

Practical Suggestions

To develop a good working relationship with data providers, consider the following:

- Try to use data that is easy for the data providers to obtain.
- Personally visit data providers whenever possible to develop a good working relationship.
- Discuss potential corrective action with data providers to emphasize the real goal of measurement.
- Never brief negative information without first informing the involved stakeholders and including their perspective in the briefing.

Ground Truth

Ground Truth is the goal of establishing project measurement archives and artifacts that are regarded by project personnel as highly reliable, comprehensive, and detailed. To attain this goal, the archives and artifacts must also be readily accessible and user friendly.

Practical Suggestion

Data should reach deep into the organization, providing a fingertip feel of the situation. See Figure 7 for an example of an approach to meet this need. Resolving integration defects on a large project requires the coordinated effort of several teams of professionals.

Defects flow through different states (e.g., review, fix, or verify) as they are processed (worked off) by different teams. If defects fail to flow as expected deep within the project, bottlenecks can occur that can slow the entire effort. With this drill-down type chart, the user can specify a bin (group of defects related to specific baselines, states, and/or teams) to be displayed and view the defect flow through the bin. The defect flow is indicated by three plots on the chart:

- **Cumulative In.** Cumulative number of defects that have entered the bin, which rises over time as new defects flow into the bin.
- **Cumulative Out.** Cumulative number of defects worked off, which rises over time as defects in the bin are

worked off.

- **Backlog.** Current number of defects being worked within the bin, which will remain constant over time if defects are worked off as fast as they flow into the bin.

The plots answer status questions for the bin such as, "Is the team keeping up with the flow of defects?" or "Why is the backlog rising?"

Conclusion

Using the measurement and analysis OODA Loop to engage a project creates a focused, fast-paced measurement process, drives corrective action, and brings measurement and analysis to its full potential. The OODA Loop also imparts a high-energy, spirited tone to measurement by keeping the focus on the project dynamics of decision making. The result is a successful and rewarding process experience.◆

References

1. Tufte, Edward R. Visual Explanations: Images and Quantities, Evidence and Narrative. Graphics Press, 1997.
2. Tufte, Edward R. The Visual Display of Quantitative Information. Graphics Press, 1983.

About the Author

Joe H. Lindley is employed by Raytheon, Intelligence and Information Systems, in Garland, Texas. He has more than 30 years of experience in software engineering, project management, process improvement, customer relations, marketing, regulatory affairs, and software measurement. His successes include development of a best practice measurement program, successful project completions, complex Food and Drug Administration medical device approvals, and a patent for Global Positioning System technology. He has a master's degree in electrical engineering.

Raytheon
Intelligence and Information Systems
1200 S Jupiter RD
Garland, TX 75042
Phone: (972) 205-7281
E-mail: joe_h_lindley@
raytheon.com

Web Sites

Center for National Software Studies

www.cnsoftware.org

The mission of the Center for National Software Studies (CNSS) is to elevate software to the national agenda, and to provide objective expertise, studies, and recommendations on national software issues. The CNSS Web site is used to form study groups, conduct research, and collaborate on study topics, and publish study results to promulgate findings and solicit feedback and participation. It recently released "Software 2015: A National Software Strategy to Ensure U.S. Security and Competitiveness."

Software Program Managers Network

www.spmn.com

The Software Program Managers Network (SPMN) seeks out proven industry and government software best practices and conveys them to managers of large-scale Department of Defense software-intensive acquisition programs. SPMN provides consulting, on-site program assessments, project risk assessments, software tools, guidebooks, and specialized hands-on training. The SPMN Web site is owned and operated by the Integrated Computer Engineering Directorate of American Systems Corporation.

Practical Software and Systems Measurement Support Center

www.psmsc.com

The Practical Software and Systems Measurement (PSM) Support Center is sponsored by the Department of Defense (DoD) and the U.S. Army. It provides project managers with the objective information needed to successfully meet cost, schedule, and technical objectives on programs. PSM is based on actual measurement experience with DoD, government, and industry programs. The Web site also has the most current version of the PSM Guidebook.

Coming Events

October 16-20
Object-Oriented Programming, Systems, Languages, and Applications Conference
San Diego, CA
www.oopsla.org/2005/ShowPage.do?id=Home

October 17-20
MILCOM 2005
Military Communication Conference
Atlantic City, NJ
www.milcom.org/2005

October 17-21
Quality Assurance Institute's 26th Annual Software Testing Conference
Orlando, FL
www.qaiusa.com

October 24-27
8th Annual
Systems Engineering Conference
San Diego, CA
http://register.ndia.org/interview/register.ndia?

November 6-9
6th Annual Amplifying Your Effectiveness Conference
Phoenix, AZ
www.ayeconference.com

November 13-17
International Association for Computing Machinery SIGAda 2005 Conference
Atlanta, GA
www.sigada.org/conf/sigada2005

November 14-18
STARWEST 2005
Anaheim, CA
www.sqe.com/starwest

May 1-4, 2006
2006 Systems and Software Technology Conference
STC Systems & Software Technology Conference
Salt Lake City, UT
www.stc-online.org

Applying Functional TSP to a Maintenance Project

Ellen George and Dr. Steve Janiszewski
PS&J Software Six Sigma

Personal Software Process^{SM} (PSP^{SM}) is taught in the context of new code development and implementation. We frequently hear developers say that it will only work for new development and that their maintenance project is different. This article dispels that myth, demonstrating how PSP and Team Software Process^{SM} (TSP^{SM}) were successfully adapted to plan and manage a maintenance project. Readers will learn the key success factors in forming a functional project team and will learn how to emphasize and de-emphasize portions of the TSP process to better address the needs of a maintenance project.

The Personal Software Process^{SM} (PSP^{SM}) is a software development process originated by Watts Humphrey at the Software Engineering Institute (SEI) in the early to mid-1990s. By design, it is a high-maturity development process with all the features required to support a single developer. PSP is a measurement-driven process that includes planning, estimating, design, personal reviews, and testing. Its basic concepts can be extended for all software development life-cycle phases.

The Team Software Process^{SM} (TSP^{SM}) was developed in the late 1990s to add team-level practices to the PSP. By so doing, the TSP makes the PSP suitable for use in a commercial software development environment. TSP begins with a facilitated project launch process that generates a detailed project plan. The project plan includes a development strategy, a tailored development process, detailed size and effort estimates, earned value (EV) plans, a schedule, a quality management plan, and a risk management plan. The launch process is a team-building exercise designed to foster a sense of ownership and commitment and to produce a high-performance work team. The TSP continues to support project execution activities via a structured weekly project status meeting and all the management practices necessary to run a full-scale development project.

The PSP for Engineers course is taught in the context of new development where the students are asked to complete a series of 10 programming assignments. As a result, there is a perception that PSP and TSP will only work for new development projects. In this article, we will demonstrate how PSP and TSP were adapted to successfully plan and manage a maintenance project.

The Team

The Maintenance Team Dilemma

Maintenance projects deal with post-development support. In general, this can include operational support and implementation of new or enhanced features as well as defect investigations and fixes. Some maintenance projects just involve working off a backlog of problem reports and change requests. These projects are relatively easy to handle in a conventional TSP launch. Although it may be necessary to modify the estimation algorithm because there is not a good correlation between effort and the size of the change, there is a known list of tasks (the backlogged problem reports and change requests) that can provide the basic input to the TSP planning process.

> *"When preparing to launch a TSP team, it is a good practice to start by identifying a successful end state. What would management, the team, and the coach consider to be a successful launch and project?"*

The situation gets more complicated when near real-time operational support is added to the mix of project tasks. The majority of project tasks may be unknown at the time of the launch because the operational anomalies have not even taken place yet. Nonetheless, the organization needs to allocate adequate resources, commit to scheduled completion dates for backlogged tasks, and manage the overall effort.

Organizations often have difficulty planning, staffing, and managing these sorts of maintenance projects due to the unpredictability of both the rate at which operational anomalies are reported and the effort required to respond to anomaly reports. There is frequently no correlation between the length of an anomaly investigation and the size of the resulting change. In fact, many anomaly reports do not even result in a change.

As a result, team members frequently feel that any attempt to estimate and plan the effort for a maintenance project is futile. And yet, these projects need to be staffed and managed.

This article shows how PSP and TSP were successfully used to plan and manage a maintenance project and how the team was able to build a useful estimating scheme in a project environment where it was common for developers to spend 75 percent of their time on unplanned event-driven anomaly investigations.

The Project

The project involved maintaining a large network-based financial services package originally implemented in C. The package was key to the company's revenue stream and required real-time operational support as well as fixes and enhancements.

The maintenance team was composed of 12 PSP-trained software developers. Half the maintenance team was located at a site in the United States, while the remaining half was located at a site in Europe. There was a team lead at each location. The project manager worked in the United States. The TSP launch was held in the team's U.S. office.

Is It a Team?

Typical work for this group of developers includes investigating reports of operational anomalies, upgrading legacy software to support new requirements, adding new features, investigating defects, and fixing defects. In the past, each developer considered himself or herself responsible for a different product. As it turned out, they were each responsible for a component of a single product. Each product component was built and released separately. There were no task or schedule dependencies between developers. Defects were generally localized to a component of the product. Up to this point, individuals only needed to be focused on

[SM] TSP and PSP are service marks of Carnegie Mellon University.

their own component.

This contrasts strongly with the situation in new development where there are usually obvious dependencies in each person's work on tasks being done by other team members as the team works together to produce a product.

Watts Humphrey defines the *functional team* as a team with the following:

> … has a functional, rather than a product, mission. While all the members may do similar work, they do not develop a single product and their individual tasks are usually quite independent. … [An] example would be a maintenance group where each member handles the repair and enhancement of a product. While several of the members might occasionally work on elements to be integrated into a common release, they would usually work alone. [1]

The developers on this project fit the definition of a functional team perfectly. Consequently, the decision was made to launch the team using the SEI's variant of the TSP launch process for functional teams, TSPf[1].

Preparation

When preparing to launch a TSP team, it is a good practice to start by identifying a successful *end state*. What would management, the team, and the coach consider to be a successful launch and project? Answering these questions requires having well-defined launch and project goals and a strategy for attaining those goals.

Questions

What were the project goals? Why was this group of individual developers being launched as a team? What did the organization or the project manager think they would be able to accomplish as a team that they would not be able to accomplish as individuals?

We drilled down on these questions in a series of launch planning meetings with the project manager. Ultimately, the project manager was able to clearly articulate three very specific objectives that provided a compelling reason for these individuals to work together as a team:

- Spread knowledge among the team and broaden experience to reduce areas of risk and increase efficiency.
- Improve real and perceived quality to reduce customer-generated interruptions.
- Increase ratio of planned tasks to reactive unplanned activities.

If they were able to accomplish these objectives, it would benefit not only the organization, but the team members as well.

Once we had a well-defined set of management objectives for the team, the next set of questions addressed the actual TSP launch. What were some ways that the team could plan for the *unplanned*, high priority interruptions that are characteristic of near real-time operational support? What might they use as a size metric? Was there a conceptual design? What does quality mean when they are modifying a small number of lines relative to the size of the base code?

The coach's goal is not to decide the answers to these questions for the team, but rather to consider some of the possible answers and to make sure that the team collected the right project data prior to the launch so they would be able to make decisions based on data during the launch.

Through this exercise of strategizing an approach for conducting the launch, it became apparent that there were two overriding themes: commonality and repeatability.

Preparing the Project Manager

Commonality was the theme that would help the individuals to gel as a team. If they found that they had enough in common with one another and that they could benefit by taking advantage of the synergy, then surely they would come together as a team.

It was the program manager's job to define the management goals that would help set the stage for the team to gel. These goals had to serve two functions: setting a long-term vision for the project and the organization while providing short-term targets to help the team focus and come together.

We decided that the best approach would be to identify a long-term vision to provide a frame of reference. To support the vision, a series of short-term goals were developed. The team would be given three to four short-term goals with the expectation that they would be able to achieve them within about a month. The first set of short-term goals was selected so that they would be achievable with relatively low risk.

Since maintenance work can be difficult to predict and plan, we decided that the team would have a mini re-launch every four to six weeks. Each of these re-launches would provide the program manager an opportunity to roll out the next set of goals on the path to achieving his overall vision.

Preparing the Team Members

A necessary ingredient to this team's success was to get the team members to believe that there was a considerable amount of *repeatability* in their work and that this repeatability would lead to an improved ability to estimate. We asked the team members to start gathering PSP data on their tasks several weeks prior to the launch in an attempt to find the repeatability in their daily or weekly activities.

We found that the developers had two distinct categories of tasks. The first category was high priority interruptions. These interrupts could occur at any time. When they did, all other work had to be put aside. The second category of task was background work, which consisted of the tasks that the developers worked on when they were not reacting to high priority interruptions.

It was obvious that the developers would not be able to anticipate what interruptions would occur. However, it seemed that there might be a pattern to the quantity of high priority interrupt effort that was spent within a period of time. If the developers could quantify the percentage of time each week that was spent handling the high priority interrupts, then they would be able to create a budget of task hours for this category of activity.

The developers were asked to flag each task they worked on as one that they either knew about at the beginning of the week or one that came in during the week as an interrupt. While the percentage by category differed by individual, we found that there was clearly repeatability for each individual from week to week. The developers were able to use the data they collected during launch meeting No. 4 to plan for time to be spent on planned tasks as well as to create a budget for time to be spent on unplanned tasks. The planned tasks would then be estimated and planned just like any other PSP task. Whenever an interrupt came in, the developer would estimate it and plan it. Time for the interrupt would be allocated from the budget for unplanned tasks.

The Launch

A TSP launch consists of a series of scripted meetings. We tailored the standard TSPf meeting scripts by employing some special techniques designed to contribute to the success of the team by reinforcing the themes of *commonality* and *repeatability*.

Tailoring – Management and Team Goals

For each goal, we asked the team why it was important to them and what the impact to the team would be if they missed it. They were being asked to justify the need for each goal. In doing so, they began to internalize the importance of the goals, converging on a common understanding of what the goals

Activity	Entry Criteria	Exit Criteria
Problem Investigation	• Anomaly report, defect report, or improvement request with assigned due date and priority.	• Root cause identified and documented. • Anomaly report closed or reassigned. • Additional defect reports or improvement requests opened as necessary. • Solution investigation assigned if needed.
Solution Investigation	• Defect report or improvement request with assigned due date and priority.	• Authorized solution.
Problem Fix	• Defect report or improvement request with assigned due date and priority. • Authorized solution (optional).	• Problem fixed. • Defect report or improvement request has test pending status.
Conformance Test	• Test request opened. • Test scripts are received from exchange.	• Test completed. • Filled scripts sent to exchange or test cancelled. • Test results published.
System Build	• System build request opened. • All defect reports and improvement requests are cross-referenced to the build request and have test past status. • All code changes in the build are traceable to a cross-referenced defect report or improvement request.	• Working system with required documentation.

Table 1: *Maintenance Process*

really meant. The team decided to break down their goals into prioritized tasks to be interleaved with their product tasks.

Tailoring – Conceptual Design

The team did not think of themselves as part of a bigger project so they were initially hesitant to spend time reviewing the conceptual design

We asked the design manager to project a system diagram on a screen and lead a discussion in which the *person most knowledgeable* with each component of the system briefed the others on the size of the component, interactions with other components of the system, and areas of risk.

This was the first time that the team members ever had the opportunity to see their own work on individual package components in an overall system context. The discussion became animated and the team members became *physically involved*[2]. In spite of their initial reticence, this was the meeting where individuals started to come together as a team.

Figure 1: *Anomaly Investigation Effort Histogram*

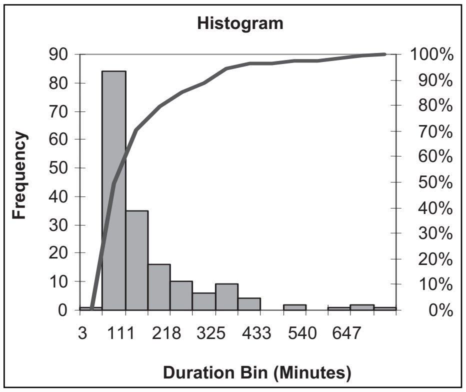


Table 2: *Anomaly Estimation Table*

	Duration (minutes)
Very Small	6
Small	18
Medium	59
Large	187
Very Large	596

Tailoring – Process Plan

The organizational maintenance process had been defined prior to the launch. So, the team focused its discussion on defining explicit entry criteria, exit criteria, and required approvals for moving from one process step to the next. The team was able to identify gaps in the defined process and to recommend modifications to address the gaps. The resulting maintenance process was relatively simple and is shown in Table 1.

The source of the problem could be a reported anomaly, a defect report, or a request of new functionality. Anomaly reports do not necessarily require product changes to be resolved. They could be operator errors, procedural issues, etc. The scope of the solution analysis depends on the size of the required change, and it could be omitted for a trivial change. Conformance testing verifies that the software will work in the exchange environment. It requires test scripts recorded from exchange feeds. Multiple changes are aggregated into a system build and released to production.

Tailoring – Top Level Plan

The team used the data they had collected over the past several weeks and analyzed the average number of task hours per week that they had each spent on *planned* versus *unplanned* tasks. Using this data, they were able to confidently plan for how many hours they could get on task each week, budgeting a percentage of those hours for unplanned or interrupt-driven tasks.

For many of the team members, approximately 75 percent of their task hours were being spent on interrupt-driven tasks. So, those team members who were achieving a total of 12 hours on task per week planned to spend three hours per week on background tasks and budgeted nine hours a week for interrupt-driven tasks.

Tailoring – Quality Plan

The team spent considerable time discussing what quality meant to their customer. A predominant concern was defects that were returned to the team by the customer after an initial fix. The team put metrics in place to explicitly track and manage the following:

- Quantity and frequency of returned *fixes*.
- Defect investigation time.
- Defect turnaround time.

Tailoring – Detailed Plan

One of the goals of this launch was to build a plan in which the team would close out all of their high priority tasks. The background tasks were sequenced so that the high-priority tasks would be completed first. Multiple EV plans were generated so that management would have visibility into the following:

- How long it would take to complete the high priority tasks given the level of interrupts that the team was experiencing.
- How much backlog work the team would complete in six weeks.
- How long it would take to work off the full backlog of tasks.

This was easily accomplished with the help of an automated scheduling tool.

Post-Mortem

The participants felt that "team synergy was improved," that they did a "good job of balancing work load," and that there was "exposure of everyone else's jobs" with "good participation and contribution from everyone." One participant summarized the experience: "This seems like the birthday of this team."

Results

Estimating

During the launch, the team was highly skeptical about their ability to estimate anomaly investigation tasks. However, they agreed to make their best estimates and then to develop an estimating algorithm from the post-launch data. Without historical data, the team estimating error was 41 percent. With the data gathered during the launch post-mortem, the team was able to reduce the estimating error to 23 percent.

Anomaly Investigation Effort

After the launch, the team collected data on the durations of anomaly investigation tasks for several months. A histogram[3] of the data, as shown in Figure 1, indicates that the duration of anomaly investigation tasks can be modeled as a random variable and that a skewed-distribution function, like a log normal, would provide a good basis for estimating the probable duration of an anomaly investigation task.

From the fitted distribution function, it is possible to determine what the expected duration of a very small, small, medium, large, and very large task would be, as shown in Table 2. The average length of an investigation task was 59 minutes, with approximately 70 percent of all investigations requiring between 18 and 187 minutes.

Estimating Algorithm

This result leads directly to a simple technique for estimating anomaly investigations that can be employed during a TSP launch. For the tasks that are part of the backlog for a special anomaly investigation, categorize each one as very small, small, medium, large, or very large and use the estimated time provided by the table. Prorate the total for all backlogged tasks to account for the unplanned investigations by using the historical percentage of time devoted to unplanned anomaly investigations. For this team, the percentage of time devoted to unplanned anomaly investigations was 75 percent to 85 percent of its available time.

Re-Estimating the Launch

During the launch, the team had estimated the effort of each identified investigation as small, medium, or large. Assessment of effort had been made by the task owner, based on familiarity with the functionality and amount of code that would need to be reviewed.

During post-mortem, the original small/medium/large estimate from the launch was used along with the calculated values of small, medium, and large to re-estimate the tasks. As shown in Table 3, the re-estimate based on the calculated size ranges reduced the estimation error by a factor of two to a total error of 23 percent.

	Actual Time	Est. Time (Launch)	Est. Time (S/M/L)
Minutes	2,549	4,301	3,313
% Error		- 41%	- 23%

Table 3: *Comparisons of Estimates*

Lessons Learned

The coach must prepare for every launch. The coach needs to understand the peculiarities of each team, anticipate potential trouble spots relative to planning, and have a strategy for how to facilitate the launch to engage the participants and to build a plan that supports the business objectives. Choose no more than three to four very specific and achievable goals to help unite the participants as a team. Use PSP 1.0 to collect data prior to the launch, and focus the team on actual data to help team members find the repeatability in their work and to make fact-based decisions during the launch.

Even though an individual task may be completely unpredictable, once the statistics that characterize a set of typical tasks are known, it is possible to use those statistics to make reasonably accurate estimates about the effort required to handle the normal workload associated with many *unpredictable* tasks. This allows a team to allocate the right number of resources to meets its commitments and bring a sense of control and predictability into an apparently chaotic project.◆

References

1. Humphrey, Watts. Coaching Development Teams. Addison-Wesley (in press to be published in 2005 or 2006).

Notes

1. TSPf is still in the prototype stage, but the SEI authorized piloting its use with this team.
2. Our experience has been that there comes a point in every successful launch where the team members get physically involved in the meetings. They sit up straighter in their chairs, leaning forward to hear better, or stand up and move chairs from the back of the room to the front so that they can see better or come forward to write on the white board.
3. The histogram shows the number of data samples falling into each of the duration bins indicated on the x-axis (bar chart) and the cumulative number of data samples with duration less than the x-axis value (curve). For example, there are 83 anomaly investigation tasks with duration between three and 111 minutes and approximately 65 percent of the data points have a duration less than 111 minutes.

About the Authors

Ellen George is a principal with PS&J Software Six Sigma and is active in training and consulting on the application of software process and project management. She has over 20 years of experience in software development, process improvement, and project management. George is a Software Engineering Institute authorized Personal Software Process℠ instructor and Team Software Process℠ launch coach. She has a Master of Science in computer science and a master's degree in technology management from the Stevens Institute of Technology.

PS&J Software Six Sigma
P.O. Box 463
Palisades Park, NJ 07650
Phone: (201) 358-8828
E-mail: ellengeorge@softwaresixsigma.com

Steve Janiszewski, Ph.D., consults on project management, Capability Maturity Model® Integration, software metrics, Personal Software Process℠, Team Software Process℠, and Six Sigma for software development for PS&J Software Six Sigma. He has 30 years of experience in all phases of software development, management, and process improvement. Prior to joining PS&J, Janiszewski was director of Honeywell's corporate Software Six Sigma organization, providing process assessments, training, and consulting to more than 100 locations around the world. He has a Master of Science and doctorate in theoretical physics from New York University.

PS&J Software Six Sigma
P.O. Box 463
Palisades Park, NJ 07650
Phone: (201) 947-0150
E-mail: stevejaniszewski@softwaresixsigma.com

Tying Project Measures to Performance Incentives

David P. Quinn
Borland Software Corporation

In the struggle to set expectations and reward good performance, organizations sometimes tie project measures to performance incentives. To avoid the pitfalls sometimes caused by this link, we must explore the nature of project measures and performance and how to possibly link project measures to incentives so the intended behavior occurs.

Any measure when viewed in isolation is open to misinterpretation and misuse. Without the full context of other measures, we may unjustly attribute good or poor performance based on viewing a single measure as an absolute. This is why linking incentives to individual measures can be detrimental to what the organization is trying to accomplish. People tend to focus on meeting a measure instead of accomplishing an outcome. People tend to focus on themselves instead of the team or the organization, causing problems for others and negatively impacting the overall organization's performance. In fact, linking incentives to measures often has some unintended consequences.

Effort Variance as an Example

Effort variance is a very useful measure when used for project management as an indicator. It can show problems with estimates, processes, project scope, and performance. These problem areas have a different meaning depending on whether effort variance is indicating less effort expended than estimated or more effort expended than estimated.

In the case of less effort expended than estimated (e.g., 100 hours estimated but only 80 expended for a –20 percent effort variance), here are some causes of why the variance may have occurred:

- Estimate
  - Engineers padded their estimates.
  - Estimation parameters were wrong.
- Scope
  - Lack of understanding of scope.
  - Scope reduced but estimate not updated.
- Process
  - Steps skipped.
  - A process improvement occurred.
- Performance
  - Hours worked were not recorded/reported.
  - Brilliant work was performed.
  - Missed one or more requirements.
  - Did not complete the task.
  - Allowed poor quality in order to meet a deadline.

Of these causes, only two (brilliant performance and process improvement) should result in a reward through an incentive. The other causes should result in some sort of remedial action.

On the other hand, when looking at more effort expended than estimated (e.g., 100 hours estimated but 120 expended for a +20 percent effort variance), here are some causes of why the variance may have occurred:

- Estimate
  - Someone lowered the original estimates to meet mandated cost/schedule.
  - Estimation parameters were wrong.
- Scope
  - Lack of understanding of scope.
  - Scope increased but the estimate was not updated.
  - Customer was indecisive on the requirements.
- Process
  - Process is inefficient.
  - The process does not match the customer's needs.
  - Unnecessary/inappropriate steps were taken.
  - The work from someone earlier in the process made this person's work more difficult.
- Performance
  - Meets initial estimate but does not meet the modified estimate to meet budget/schedule (i.e., someone changed the estimate without changing scope just to make the numbers match the preferred schedule).
  - Poor work performance.
  - Added capability the customer did not request.

> ***"The biggest potential problem with linking incentives to effort variance is that people may game the system."***

As with the previous example, not all of these causes should detract from incentives. Only poor work performance and using the wrong parameters are signs that the engineer needs to adjust his/her behavior. Many of the others are organizational or managerial faults that impact the engineer's performance.

Before rewarding or punishing for effort variance, there are other factors to consider. Perhaps it is okay that effort variance is above estimate if the quality of code leads to reduced effort variance for testing (i.e., there is no real impact to schedule) or reduced rework to correct defects. Perhaps the minor changes in scope (that no one felt required changing in the estimates) really hit harder than thought and showed up in the effort variance. You would not know if the latter case was true unless requirements volatility measures were gathered.

Other Indicators

The impact of effort variance on projects and on the organization must be examined to get an adequate picture of the importance of effort variance. For instance, a team of developers may take shortcuts to reduce effort variance, but the shortcuts negatively impact quality. That may increase the effort variance of the test team who must perform more test cycles than estimated. The developers may get rewarded for their improved effort variance at the expense of the test team.

Significant effort variance should result in a change in schedule performance. If team members' effort variance is –20 percent, the schedule should see a comparable variance (i.e., the schedule variance should be roughly –20 percent). When there are major gaps between the two variances (for example, –15 percent effort variance and –5 percent schedule variance), this should signal that something may be wrong. The organization needs to investigate why the effort per-

Measure	High	Medium	Low	Unsatisfactory
Effort Variance	Any variance better than -X percent.	Variance between -X percent and -Y percent.	Variance between -Y percent and +Z percent.	Variance greater than Z percent.
Schedule Variance	Any variance better than -X percent.	Variance between -X percent and -Y percent.	Variance between -Y percent and +Z percent.	Variance greater than Z percent.
Quality	Any variance better than X percent of the team average.	Variance between X percent and Y percent of the team average.	Variance between Y percent and Z percent of the team average.	Defect rate worse than Z percent of the team average.
Process Compliance	Recommended process improvement accepted for implementaion and complied with defined processes.	Participated in process improvement activities and complied with defined processes.	Complied with defined processes on regular basis.	Inconsistent use of defined processes.

Assumptions X,Y, and Z may be different for each measure.
Quality may move from being relative to team members to being organizationally relative.
More measures could be added if desired.

Table 1: *Example Performance Incentive Structures*

formance is not impacting the schedule more. Perhaps other factors inhibit an improvement in schedule variance.

Other Problems With Linkage

There are other problems with linking incentives to individual measures. Based on the organizational learning that occurs from doing projects, performance that once provided a bonus for people will no longer result in a bonus as the organization adjusts process performance measures. The following provides an example for effort variance.

One of the reasons to gather effort information is to ensure your estimating model is correct. If team members show they are expending far fewer hours than estimated on a regular basis (better than –10 percent effort variance), the organization should update the estimation model to reflect this performance. This will provide more accurate effort estimates in the future. However, once the organization modifies the estimation model, team members will no longer qualify for incentives, even though they perform at the same level the organization rewarded before.

But not updating the estimation model is wrong. If the customer continues to see that effort variance is significantly below estimates, the customer is likely to assume that engineers are padding their estimates. Trust is broken and the customer will require that the organization reduce its effort estimates on future projects no matter how reasonable the estimate may be. The organization must update the estimation model despite the impact on the incentives.

Performance Is Relative

Just as one measure by itself may not tell the whole story, sometimes that one measure hides the truth. While other measures may give a more accurate picture of where performance stands as a whole, measures may indicate where an individual engineer stands against his/her peers. A single measure by itself may indicate that an engineer is performing well, but in comparison to other engineers, the engineer's performance may be lacking.

While it may appear logical to reward someone who has a –10 percent effort variance, it does not make sense if the organization's average effort variance is –15 percent. Technically, this person's performance is slowing down the organization, and this person is not performing up to the level of his/her peers. Likewise, it may be good to reward someone with a +5 percent effort variance if the rest of the organization is performing at a +12 percent effort variance.

There is a problem with this comparison though. The engineer with a –10 percent effort variance while everyone else has a –15 percent effort variance may be handling all the tough, complex tasks. Comparing performance between peers using measures like effort variance is not as wise as it may appear.

Other Potential Problems

Someone who consistently outperforms the estimated effort does something different from everyone else. If that person does not share that something different with other people, the incentives are reinforcing the wrong behavior. As someone discovers a process improvement or other type of improvement, the person should share that information, and the organization should reward that sharing. Someone who does not share improvement information is not acting in the organization's best interest.

The biggest potential problem with linking incentives to effort variance is that people may game the system. This usually takes two forms: people work extra hours but do not record them, or people place the hours worked against a different activity. In the latter case, these hours are not necessarily charged to non-billable activities. People may charge the hours to activities on other projects with available budget.

In either case, the organization is not getting accurate information on what it takes to get a project done. The organization is no longer able to learn how accurate the estimates are or if a process improvement occurred. Organizations that find people gaming the system because of the linkage between incentives and measures usually have to take one of two solutions. The first is to remove the linkage. The second is to have a policy that makes entering inaccurate effort data a cause for dismissal. Organizations tend to choose the former rather than the latter because the organization does not want to lose good people.

Recognizing and Rewarding Outstanding Performance

Project performance measures can be a contributor to recognizing outstanding performance, but there should not be a direct correlation between a single measure and an incentive. A measure that makes up X percent of an incentive (no matter how small X is) is more likely to be gamed, destroying any chance to accurately measure organizational performance. A better approach would be to group a number of measures together and reward them as an overall performance.

For instance, an organization may rate a software engineer on effort, schedule, quality, and process compliance (see Table 1). The organization may rate each factor as high, medium, low, or unsatisfactory (see Table 2). The combination of highs,

Table 2: *Compensation Formulas*

Highs	Mediums	Lows	Unsatisfactory	Percent Compensation
4	0	0	0	**100%**
3	1	0	0	**95**
3	0	1	0	**90**
2	2	0	0	**90**
2	1	1	0	**80**
2	0	2	0	**75**
1	3	0	0	**75**
1	2	1	0	**60**
0	4	0	0	**50**
1	1	2	0	**50**
0	3	1	0	**25**
1	0	3	0	**25**
0	2	2	0	**15**
0	1	3	0	**10**
0	0	4	0	**0**
-	-	-	1 or more	**0**

mediums, and lows indicate the incentive reward. Getting a high rating in all four factors would get the maximum incentive. Getting three high ratings and one medium rating would get a certain percentage. The organization would further adjust incentives for different combinations of highs, mediums, and lows. Any unsatisfactory rating would automatically eliminate the incentive.

This helps the engineer understand that each of these measures is important but no one measure is more important than the other measures. It also allows for the fact that the measures are interdependent and focuses on how the organization views technical performance as a whole.

Personal Experience With Unintended Consequences

I have two sons, David and Mark. Like most young boys, they constantly want raises in their allowances. I am a firm believer that an increase in wages results from an increase in responsibility. I also am a firm believer in trying to get the boys to do chores I do not find entirely enjoyable. Therefore, I needed to devise a way to allow them to earn more money while making my life easier.

I targeted mowing the lawn, a chore I dislike. Unfortunately, they were too young to mow the lawn. However, before I mow the lawn each week, there is another necessary chore. We have two dogs: Ace, a retired racing greyhound, and Amigo, a longhaired Chihuahua. Their daily routine generates a set of piles that someone must gather before I can mow the backyard. That became their chore.

David, the older son, gets $2 for cleaning the piles in the backyard. When David is done, Mark gets $1 to find any missed piles. However, I add an incentive to this. For every Ace pile Mark finds, Mark gains 25 cents and David loses 25 cents. For every Amigo pile Mark finds, Mark gains 10 cents and David loses 10 cents. David cannot go below $1 total but Mark has a limitless incentive. When Mark finishes his chore, I do a final inspection of the backyard. As with David, Mark loses 25 cents for every Ace pile I find and loses 10 cents for every Amigo pile I find. It seemed like a great system.

One spring I sent David out to do his task. I lost track of Mark shortly after David started but found him shortly before David was done. Since it was starting to get dark, I started Mark on his chore while David finished the last third of the yard.

As they both continued their chores, David suddenly announced that one of the Ace artifacts was covered with grass. I did not think much of it until he announced a few seconds later that other artifacts were covered by grass. Mark commented that more artifacts are likely covered by grass. I was able to put two and two together and pointedly asked Mark if he had been covering up the artifacts. His face turned white, his jaw dropped, and he meekly let out a "Yes."

I immediately sent him in the house to get his bath and go to bed. I also let him know he forfeited his money for doing the chore. As soon as the door closed, I laughed so hard I thought I would cry. I did not realize my performance incentives would create that type of behavior. You really have to be careful when linking performance measures to incentives.

Conclusion

Gathering and using measures is an essential part of business. Measures provide outstanding insight into current status, possible problems, and process improvements. However, one measure does not provide enough insight by itself. Measures must be viewed in total as the result of one measure may impact another measure. The data gathered for the measures must be unquestionably accurate. When organizations tie incentives to project measures, people often report the data inaccurately to meet the incentives. Once organizations realize data is inaccurate, they tend to break the link between incentives and measures. They recognize that data needs surpass the incentive needs.◆

About the Author

David P. Quinn is a principal consultant with Borland Software Corporation. He has more than 20 years of systems and software engineering experience. His past positions ranged from software developer to project manager to mid-level manager as well as Engineering Process Group leader. Quinn is an authorized Lead Appraiser and past member of the Software Engineering Institute's Capability Maturity Model® Advisory Board.

30 Quail Ridge RD
Hanover, PA 17331
Phone: (717) 632-1047
E-mail: david.quinn@borland.com

A Model About Nothing

Welcome to "Backfire," the interview within a journal where we cross-examine popular icons for software truth. This month we have the cast of Seinfeld – Jerry, George, Elaine, and Kramer who have developed a Comedy Maturity Model (CMM) to teach young comedians.

Gary: First things first, what gave you the idea to teach comedy to the masses?
Jerry: Everybody is so serious now days. I thought to myself, what is going on in this community? Are you people aware of what is happening? What is driving you to this behavior? Is it the humidity? Is it the Muzak? Is it the white shoes?
George: That is so true! I have no funny friends. I am the funny one – El Clowno!

Gary: So you formed a company.
Kramer: Yes, the company is Somewhat Comical Institute (SCI), pending legal review.
Jerry: I preferred the name Super Silly Inc.
George: Elaine wanted the name to be Silly Putty Limited, for obvious reasons, but there were trademark issues.

Gary: Who came up with the idea of creating a Comedy Maturity Model?
Jerry: Kramer. He was watching a PBS special on software.
George: I love those people. You can't ask them questions. They are so mentally gifted that we must not disturb the delicate genius unless it is in the confines of an office. When huge sums of money are involved then the delicate genius can be disturbed.
Kramer: I come up with these things, I know they are gold but nothing happens, hence, I called Elaine.

Gary: Why Elaine?
Kramer: She's a calculating, cold-hearted businesswoman. When there is dirty work to be done, she doesn't mind stomping on a few throats.

Gary: Why not George or Jerry?
Kramer: There is a little too much chlorine in that gene pool.
George: Yeah, I'm a great quitter. It's one of the few things I do well. I come from a long line of quitters. My father was a quitter; my grandfather was a quitter ... I was raised to give up.

Gary: Tell me about the model.
Elaine: It has five levels. All good models have five levels.
George: I wanted seven.
Jerry: The first level is "Breathing," everyone qualifies in this level because we believe everyone has a little humor.
George: I say stupid things all the time. I cannot go two minutes without saying something stupid.
Kramer: We didn't want to exclude potential clients so we set the bar low. It's like soccer – everyone gets to play.

Gary: What comes after the breathing level?
Elaine: Level 2 is "Grin," the basic art of remembering jokes. Level 3 is "Giggle," delivering jokes. Level 4 is "Guffaw," the art of creating jokes. Level 5 is "Gut Buster," stringing jokes into a witty routine.
George: Each level has several KCAs.

Gary: KCAs?
Jerry: Knowledge Comedy Areas.
Kramer: We load the lower levels with a lot of banal KCAs and then reduce them to a few pedantic KCAs in the upper levels. That allows us to hook them and keep them.

Gary: There is not much comedic meat to the model?
George: I think that you think that a certain something is not all that it could be, when, in fact, it is all that it should be ... and more!
Jerry: You have to be patient with models; it is like knocking over a Coke machine. You can't do it in one push. You have to rock it back and forth a few times, and then it goes over.
Elaine: You know what they say, "You don't sell the steak; you sell the sizzle."

Gary: It sounds like the SCI CMM is more about making money than helping comedians.
George: Why would we want to help somebody? That is what nuns and Red Cross workers are for.
Kramer: It's like the Dewey Decimal System ... what a scam that was. I could raise enough money to cure polio.
Jerry: It's about nothing, a model about nothing.

Gary: How will you make money on the CMM?
Kramer: Assessments, workshops, and consulting.
Gary: Cosmo Kramer is consulting?
Kramer: Oh, I'm out there Gary, I'm out there!

Gary: Who would consider you comedy consultants?
George: Hey, we have artistic integrity.
Jerry: Artistic integrity? Where did you come up with that? You are not artistic and you have no integrity.
George: You know, if you take everything I have done in my entire life and condense it down into one day, it looks decent.

Kramer: The real money is in the spin-offs baby!
Gary: Spin-offs?
Elaine: Oh yes we have the Stand-up Comedy Maturity Model (Sup-CMM), Improvisation Comedy Maturity Model (I-CMM), Situation Comedy Maturity Model (Sit-CMM), Political Comedy Maturity Model (P-CMM), and for the night owls the Late Night Comedy Maturity Model (Ln-CMM).

Gary: Will all the spin-offs confuse young comedians?
Jerry: Of course, but we will integrate the models and sell services to understand the new integrated model.

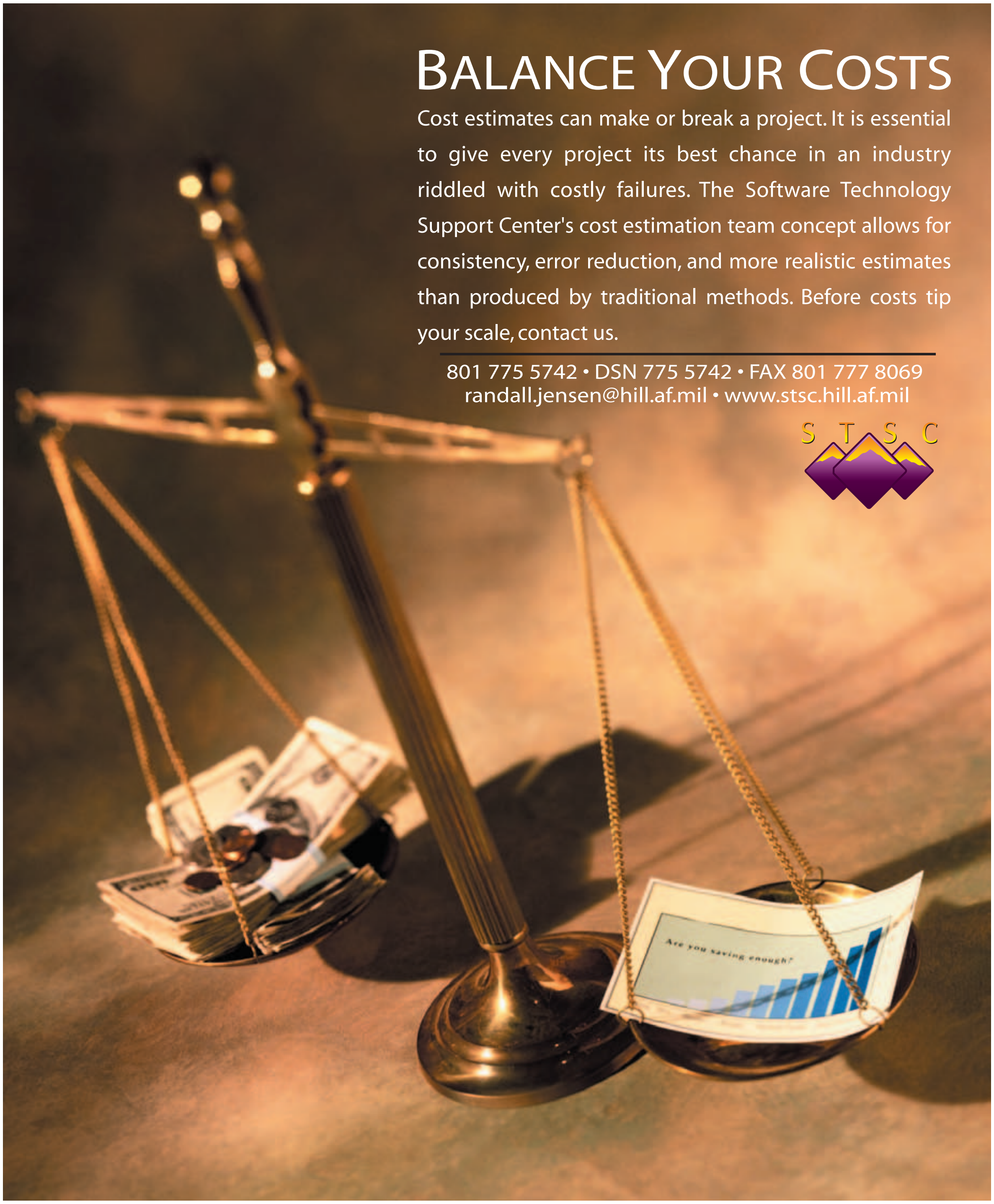
Gary: It seems like a long road just to get back to one model.
Jerry: The road less traveled is less traveled for a reason.
George: It's like selling them a car, you stick them with the undercoating, rust proofing, dealer prep ... suddenly they are on their backs like turtles.

Gary: So, it is a lie.
George: Just remember, it's not a lie if you believe it.

Gary: You're crazy.
Kramer: Are we, or are we so sane that you just blew your mind?
Jerry: Maturity models are funny business.

Okay, thank you Jerry, George, Elaine, and Kramer. Join us next time when Backfire cross-examines Butch Cassidy and the Sundance Kid on outsourcing.

– Gary A. Petersen
Shim Enterprise, Inc.
gary.petersen@shiminc.com



Software Engineering Division
Ogden Air Logistics Center

Co-Sponsored by
U.S. Air Force
Air Logistics Centers
MAS Software Divisions

CROSSTALK / 309 SMXG
6022 Fir AVE
BLDG 1238
Hill AFB, UT 84056-5820

PRSRT STD
U.S. POSTAGE PAID
Albuquerque, NM
Permit 737