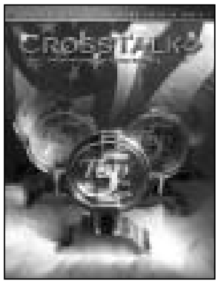
SPECIAL QUALITY SOFTWARE AWARDS ISSUE

CrossTalk

July 2003 The Journal of Defense Software Engineering Vol. 16 No. 7

Top Five Quality Software Projects



On The Cover
Cover Design by Kent Bingham.

Software Engineering Technology

Open Forum

Online Articles

Departments

CrossTalk

Sponsor	*Lt. Col. Glenn A. Palmer*
Publisher	*Tracy Stauder*
Associate Publisher	*Elizabeth Starrett*
Managing Editor	*Pamela Bowers*
Associate Editor	*Chelene Fortier-Lozancich*
Article Coordinator	*Nicole Kentta*
Creative Services Coordinator	*Janna Kay Jensen*
Phone	(801) 586-0095
Fax	(801) 777-8069
E-mail	crosstalk.staff@hill.af.mil
CrossTalk Online	www.stsc.hill.af.mil/crosstalk
CRSIP Online	www.crsip.hill.af.mil

Subscriptions: Send correspondence concerning subscriptions and changes of address to the following address. You may e-mail or use the form on p. 34.

Ogden ALC/MASE
6022 Fir Ave.
Bldg. 1238
Hill AFB, UT 84056-5820

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

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

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

STSC Online Services: www.stsc.hill.af.mil
Call (801) 777-7026, e-mail: randy.schreifels@hill.af.mil

Back Issues Available: The STSC sometimes has extra copies of back issues of CrossTalk available free of charge.

The Software Technology Support Center was established at Ogden Air Logistics Center (AFMC) by Headquarters U.S. Air Force to help Air Force software organizations identify, evaluate, and adopt technologies to improve the quality of their software products, efficiency in producing them, and their ability to accurately predict the cost and schedule of their delivery.

Top 5 Winners' Technologies Aim to Support the Warfighter: Several Used in Operation Iraqi Freedom

Serving as deputy director for Software-Intensive Systems in the Office of the Under Secretary of Defense Acquisition, Technology and Logistics Defense Systems Directorate, I spend much of my time addressing software-related issues. Thus, I am grateful to take this opportunity to publicly recognize software successes by acknowledging our 2002 U.S. Government's Top 5 Quality Software Projects.

I found it interesting that all the winners were functions of the Department of Defense, especially since the winners were selected in February before Operation Iraqi Freedom started. I can visualize how they contributed to the success our United States forces achieved. The soldiers used the Enhanced Position Location Reporting System to locate the position of both enemy and friendly troops, and to then communicate those positions to other friendly troops. The Joint Helmet Mounted Cueing System was also deployed to Iraq; its specific use is still not publicized at this time. The One Semi-Automated Forces Testbed Baseline is not currently deployed, but is planned for use in possible future conflicts. While the radars supported by the Kwajalein Modernization and Remoting software were not directly a part of the recent coalition effort, their open system architecture does provide the possibility of their use in future operations. During any conflict, our hearts and prayers are with the soldiers fighting the war; however, the civilians at home still play a crucial supporting role to the soldiers, and the Defense Civilian Pay System is an important part of how we thank these behind-the-scenes workers.

This issue of CrossTalk presents an inside look at how these systems were developed and what made them successful. I am hoping that other organizations developing software for the United States government will consider the merits of these projects and emulate their best practices in the projects they develop for our government. Since the United States depends on the software that enables our systems, our country will continue to become stronger as the quality and capabilities associated with software continue to improve.

In addition to the articles discussing the winning Top 5 projects, this issue of CrossTalk includes articles that address processes for developing quality software. The first supporting article by Dennis R. Goldenson, Terry Rout, and myself, *Measurement and Analysis in Capability Maturity Model Integration Models and Software Process Improvement*, stresses the need for measurement and analysis in the software development process and discusses how this is now stressed in the Capability Maturity Model Integration.

The next article, *Combat Resistance to Software Measurement by Targeting Management Expectations*, by Carol A. Dekkers, takes the discussion of measurements a step further by providing insights into implementing and sustaining a measurement program within an organization. From my perspective, if you prefer using the term metrics instead of measurement, then I recommend using it as an acronym: **M**easure **E**verything **T**hat **R**esults **I**n **C**ustomer **S**atisfaction. This way, perhaps next year your project will be recognized as one of the Top 5, for customer satisfaction is the ultimate measure of project success.

Lastly in this issue, Sarah A. Sheard from the Software Productivity Consortium provides sound advice in her article, *Life Cycle of a Silver Bullet*.

While the processes implemented by this year's Top 5 winners are examples of how to succeed, they should not be twisted into a quick fix for ailing projects. First, project members need to look within their organizations for their own strengths and challenges, then use this information as leverage for improving their own projects. If you are doing good work and would like to share your success with others, I hope you will consider submitting a nomination for next year's Top 5 award.

Joe Jarzombek, PMP
Deputy Director for Software-Intensive Systems Defense Systems Directorate
Office of the Under Secretary of Defense (Acquisition, Technology and Logistics)

Software Project Winners Exemplify Software Development Best Practices

Elizabeth Starrett
CrossTalk

CrossTalk is proud to present this series of articles featuring the winners of the 2002 U.S. Government's Top 5 Quality Software Projects. These top five projects were selected from 70 nominations in this second annual event. They demonstrate how competent software project teams go about building successful products. In the following pages, we present a brief article on each winning project, along with biographies of the judges, and brief summaries of the top nine finalists.

U.S. Government's Award-Winning Quality Software Projects

The results are in, and it is clear that the government is building many successful software packages that are top-notch examples in project management and quality control. Listed below in alphabetical order are the winners of the second annual 2002 U.S. Government's Top 5 Quality Software Projects contest managed by CrossTalk. We congratulate them and hope you enjoy reading more about their winning projects in the following pages.

- **Defense Civilian Pay System**
  Customer Organization:
  Department of Defense
  Location: Pensacola, Fla.
- **Enhanced Position Location Reporting System**
  Customer Organization: U.S. Army, CECOM, PM TRCS
  Location: Fullerton, Calif.
- **Joint Helmet Mounted Cueing System Software Upgrade**
  Customer Organization: Joint Air Force/Navy Program Office, ASC/FBH Wright-Patterson AFB
  Location: Wright-Patterson AFB, Ohio
- **Kwajalein Modernization and Remoting**
  Customer Organization: Space Missile Defense Command/U.S. Army Kwajalein Atoll/Reagan Test Site
  Location: Lexington, Mass.
- **OneSAF Testbed Baseline**
  Customer Organization: Program Executive Office – Simulation, Training, and Instrumentation
  Location: Orlando, Fla.

Software affects the lives of most Americans. One example of this evidence is its use as an enabler by branches of the U.S. government. Software plays a part in communication, traffic control, U.S. census, and the Department of Defense just to name a few.

This importance is what prompted the Office of the Under Secretary of Defense (OSD) and the CrossTalk staff to sponsor the second U.S. Government's Top 5 Quality Software Projects. Dr. Nancy Spruill, former director, Acquisition Resources and Analysis, announced the contest in the September 2002 issue of CrossTalk. We are again honored by the status of the judges, including Dr. David A. Cook (Air Force), Carol A. Dekkers (industry), Robert Gold (OSD), Capers Jones (industry), Edward C. Thomas (Army), Dr. Gerald M. Weinberg (industry), and Brenda Zettervall (Navy).

The OSD sponsored this award to recognize software best practices within the U.S. government. Information from the winners and finalists is being leveraged as examples of the right way to develop software. Winners have been asked to share their nomination information with other leaders in the software community; are invited to attend the Amplifying Your Effectiveness Conference to share their expertise with others; and CrossTalk's parent organization, the Software Technology Support Center, arranged for the winners to discuss their projects and answer questions at the 2003 Software Technology Conference. Finally, CrossTalk developed this special issue to share these successes.

CrossTalk received 70 nominations for this year's award. Each project nomination, along with customer feedback, was reviewed by at least three STSC software professionals against a set of established criteria (see page 17). The users of the software that received the best overall scores in the first round of scoring were contacted for their feedback on the software. These projects were then reviewed again and reduced to 14 finalists. These nominations, along with the customer and user feedback, were sent to the final board of judges who made the final selection of the Top 5 winners. The selection was not easy, since all of the finalists had their own special merits that had to be considered.

The winning projects were presented with their awards at the Software Technology Conference on April 29, 2003 in Salt Lake City. This award was presented by Joe Jarzombek, the deputy director for Software Intensive Systems at the OSD, in front of top ranking officials from each of the military services.

Pattern of Success

The 70 nominations featured software that conserves resources by being implemented across multiple platforms, improves morale, and saves lives. An Air Force user of one of the finalist's projects states, "The improvement was so great, that when the Navy personnel evaluated it, they immediately recommended to the Navy that they adopt this product rather than build their own. The Navy then approved its [the software's] use ... The Army has realized that this contractor did such an excellent job in the development of this product, that they now have a dialogue to address similar Army problems."

A customer of another finalist stated, "[The software is] saving hours per day off the maintenance cycle, which allows a few more hours for sailors to eat, sleep, relax, write home, etc., reducing overall fatigue and increasing general morale."

Another user stated, "More sons and daughters will be coming home as a result of the software."

"Having scored for two years, the patterns of success are becoming clear," said Capers Jones. "The overall results are adding solid empirical findings about the efficacy of peer reviews, defect prevention, and the value of climbing above Capability Maturity

Continued on page 33

Top 5 Quality Software Projects Judges' Biographies

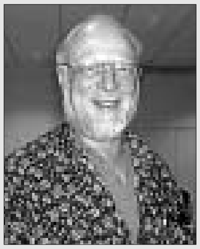
Dr. David A. Cook is the principal engineering consultant at Shim Enterprise, Inc. He is currently assigned to the Software Technology Support Center, Hill Air Force Base, Utah. He was formerly associate professor of computer science and also department research director at the U. S. Air Force Academy, and also a former deputy department head at the Air Force Institute of Technology. He was a member of the Air Force Ada 9x Government Advisory Group, and has published numerous articles on software process improvement, software engineering, object-oriented software development, and requirements engineering. Cook has a bachelor's degree in computer science from the University of Central Florida, a master's degree in teleprocessing from the University of Southern Mississippi, and a doctorate in computer science from Texas A&M University. <david.cook@hill.af.mil>

Carol A. Dekkers is president of Quality Plus Technologies, Inc., a management consulting firm specializing in improving software processes, including software measurement, software quality, process improvement, requirements, and software sizing. Dekkers is a certified management consultant, certified function point specialist, professional engineer, and information systems professional. She is currently the chair of the Project Management Institute Metrics Special Interest Group, the annual Quality Congress Track chair for the American Society for Quality (ASQ) Software Division, and a project editor on behalf of the United States to ISO's software engineering standards subcommittee. ASQ's Quality Progress named Dekkers as one of the 21 New Voices of Quality for the 21st Century for her vision of software quality. She has presented internationally at conferences and has published articles in leading industry journals. <dekkers@qualityplustech.com>

Robert Gold is the assistant for acquisition, Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics, Strategic, and Tactical Systems. He has almost 20 years of experience in the acquisition and sustainment of software-intensive systems. Gold is currently responsible for development and maintenance of all Department of Defense policy related to the acquisition of software-intensive systems and related education initiatives for the acquisition workforce. He is currently on rotational assignment from the Naval Sea Systems Command where he has served in a variety of positions. Gold is a member of the Defense Professional Acquisition Workforce and is Level 3 certified in both Systems Engineering and Program Management. He has a Bachelor of Science in electrical engineering from Lehigh University, a Masters of Science in systems engineering from Virginia Tech, and a Public Management Certificate from Indiana University-Purdue University, Indianapolis, Ind. <rob.gold@osd.mil>

Capers Jones is chief scientist of Artemis Management Systems and director of Software Productivity Research Inc., Burlington, Mass. Jones is an international consultant on software management topics, a speaker, a seminar leader, and author. He is also well known for his company's research programs into critical software issues: Software Quality: Survey of the State of the Art; Software Process Improvement: Survey of the State of the Art; Software Project Management: Survey of the State of the Art. Formerly, Jones was assistant director of programming technology at the Illinois Institute of Technology Programming Technology Center in Stratford, Conn. Prior to that he was at IBM for 12 years. Jones received the IBM General Product Division's outstanding contribution award for his work in software quality and productivity improvement methods. <cjones@spr.com>

Edward C. Thomas leads Communications Electronics Command's efforts to provide state-of-the-art software engineering products and services throughout the Army and the Department of Defense. These products and services include enterprise-level software architecting and integration; software technology assessment and application; system-level software engineering for more than 400 individual Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance and business systems; and worldwide technical support to war fighting units. He leads a global organization of more than 3,000 military, civilian, and industry employees and manages an annual budget of approximately $300 million. Thomas has worked in various capacities for the Army since 1974 and was appointed to the Senior Executive Service in 2001.



Gerald (Jerry) M. Weinberg is a principal in the consulting and training firm Weinberg & Weinberg with more than 45 years experience. Weinberg is author or co-author of more than 40 books, including "The Psychology of Computer Programming," and "An Introduction to General Systems Thinking." His books cover all phases of the software life cycle, including "Exploring Requirements," "Rethinking Systems Analysis and Design," "The Handbook of Walkthroughs," "Inspections and Technical Reviews," "General Principles of System Design, and "The Roundtable on Project Management." His books on leadership include "Becoming a Technical Leader," "The Secrets of Consulting," "More Secrets of Consulting: The Consultant's Tool Kit," "The Roundtable on Technical Leadership," and the "Quality Software Management" four-volume series. Weinberg is also known for his conferences for software leaders, including the Amplifying Your Effectiveness Conference. <hardpretzel@earthlink.net>

Brenda Zettervall is a computer scientist at the Naval Surface Warfare Center, Dam Neck. She has more than 25 years of experience in the field of computer systems and software engineering for complex, real-time command and control systems used both for deployment on U.S. Navy surface ships and for simulated land-based integration. Currently, Zettervall is deputy director for the Naval Collaborative Engineering Environment, the office of the Chief Engineer for the Navy for Research, Development, and Acquisition. In addition, Zettervall provides naval coordination for software-related activities for the Department of Defense Software Intensive Systems Steering Group. She was awarded the Navy's Meritorious Civilian Service award for her work with the Office of the Under Secretary of Defense. Zettervall has a bachelor's degree in mathematics from Radford University. <zettervallbt@navsea.navy.mil>

Defense Civilian Pay System Streamlines Payroll System Operations

Chelene Fortier-Lozancich
CrossTalk

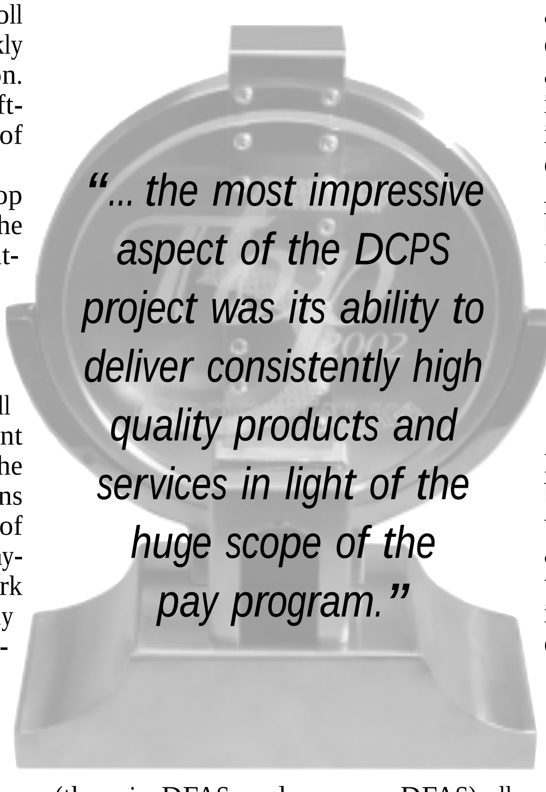
The Defense Civilian Pay System project's mission is to provide payroll processing and support to approximately 681,000 Department of Defense civilian employees biweekly. The project supports three centralized payroll offices representing total gross biweekly disbursements in excess of $1.5 billion.

The Defense Civilian Pay System (DCPS) project's mission is to provide payroll processing and support to approximately 681,000 Department of Defense civilian employees biweekly. The project supports three centralized payroll offices representing total gross biweekly disbursements in excess of $1.5 billion. The DCPS mission includes payroll software support to the Executive Office of the President.

Being named a U.S. Government Top 5 award winner is no small thing. For the DCPS, the technical expertise, commitment to success, and unceasing determination in providing consistently high quality while revamping and maintaining Department of Defense (DoD) payroll processing is no small thing. After all, when you work, you want to be paid accurately and on time. The DCPS – a large scale project with millions of lines of code and tens of thousands of users – successfully streamlined the payroll function for the DoD federal work force and achieved the Capability Maturity Model® (CMM®) Level 4 software quality standard in the process.

"We replaced much of our antiquated pay system with DCPS in 1995," says the director of the only non-Defense Finance and Accounting Service (DFAS) payroll office. DCPS' purpose in implementing a new pay system was twofold: (1) provide improved information processing services for civilian pay for the DoD with consistent implementation of pay policies, and (2) reduce costs.

Prior to DCPS implementation, civilian pay for the DoD was handled by approximately 2,800 payroll clerks with each clerk maintaining 500 pay accounts. Civilian pay in the DoD was supported by 26 different legacy pay systems operating at more than 350 payroll offices throughout the world. Pay-related services were handled within and by the various military services. As a result, pay policies for the DoD were implemented inconsistently across the multiple pay systems and at high costs due, in part, to duplication of effort. The DoD now pays this same customer base using four payroll offices (three in DFAS, and one non-DFAS) all supported by one system – DCPS. Today, DoD civilian pay is supported by approximately 300 payroll clerks; each clerk averages 2,500 accounts – a 400 percent productivity improvement. DCPS provides payroll services for the Executive Office of the President and the entire DoD – more than one-third of all civilians in the Executive Branch of the federal government.



"... the most impressive aspect of the DCPS project was its ability to deliver consistently high quality products and services in light of the huge scope of the pay program."

Arguably, the most impressive aspect of the DCPS project was its ability to deliver consistently high quality products and services in light of the huge scope of the pay program. DCPS provides payroll processing and support to approximately 681,000 federal civilians (with over 20 employee types ranging from white collar workers to teachers to firefighters) located at 1,266 activities worldwide across 13 time zones and 124 countries, and has approximately 37,000 total active online user accounts at any given time. The overall scope and success of the DCPS project (in terms of degree of standardization achieved, the number of customers/sites implemented, and the resulting cost savings to the DoD) is unprecedented in the defense financial systems community. According to annual studies conducted by Gartner, Inc. for the last six years, DCPS has been 100 percent to 300 percent more productive at half the cost than private industry averages.

The DCPS project was the first DFAS project to attain the Software Engineering Institute's CMM Level 2 in 1996, and Level 3 in 1999, according to Roger Meece, DCPS project team member. As of November 2002, DCPS was the first and only DFAS project to achieve CMM Level 4, making it one of the 7.3 percent of all assessed U.S. organizations – both private industry and government – to have been rated at CMM Level 4, i.e., at a high maturity software engineering level.

Large Project, Smooth Transition

In order to achieve consistent implementation of pay policies and desired cost savings from consolidation of the DoD civilian pay function, the DCPS System Manager Office (SMO) early on identified and addressed two key objectives: (1) how to efficiently and effectively incorporate the functionality required to service new DoD accounts as they were consolidated, and (2) how to improve the automated system itself as well as increase the productivity of the payroll office technicians, customer service representatives, and timekeepers who are the primary users of the system.

A well-defined, repeatable, and enforced implementation plan for account consolidations ensured timely

® Capability Maturity Model and CMM are registered in the U.S. Patent and Trademark Office.

arrival of new or customer-specific functionality as customer accounts were added. This implementation plan was the cornerstone of the civilian pay consolidation effort and was used extensively during the implementation of DCPS across the DoD.

The second objective was addressed by establishing a strong change management process within the program in conjunction with a focused effort to identify, incorporate, and/or build civilian pay *best practices* into the DCPS system itself, including improvements such as automation of retroactive processing, interfacing personnel actions, and retirement. Further, the change management process addressed the need to balance and resolve DoD priorities with the priorities of the various DCPS customers.

Methods of Ensuring Quality

DCPS subject matter experts had extensive interaction with customers in the area of requirements gathering. Functional and technical staff members regularly met face to face during the requirements phase of the software development cycle. Through these detailed and personal discussions, requirements were clearly defined and documented, virtually eliminating any misunderstanding of what the customer requested and expected to receive.

Peer reviews, turnover reviews, software quality assurance, a formal test process, risk management, and statistical process control were used to ensure product, process, and project quality. Reviews were held within every software engineering phase to identify and remove defects. Review issues were continuously examined for trends, improvement, and defect cause and effect. Formal turnover reviews were held between engineering phases to ensure work product phase transferability.

Since DCPS is a Federal pay system, much of the change request workload is non-discretionary in nature: it is driven by the need to comply with legislative, regulatory, and/or top-driven policy changes. Although the DCPS change-management process has significantly matured over the years, the close working relationship between the civilian pay functional proponents (e.g., the military services, payroll offices, and functional requirements analysts) and the DCPS project staff stands out as a primary reason for the program's success, especially in its handling of system changes. This maturation has been particularly evident in dealing with the business-based/discretionary change requests that come from the user community. Customers participate in the review and prioritization of change requests. The change process itself (from customer submission of change requests to customer acceptance testing) is defined, documented, and understood. DCPS receives an average of 640 change requests from customers in a typical year. Changes to the system result largely from new legislation or policies, cost-based performance enhancements, and new systems features/modifications to existing functionality.

Challenges Presented

DCPS' in-progress reviews and testing efforts have historically been very effective in maintaining high quality as reflected in low defect rates. DCPS uses the *waterfall* development methodology that includes program unit testing, string testing, integration testing, and acceptance testing for every release.

Risk management is very important to the DCPS project both in identifying risks for specific change requests and/or releases, and in developing and tracking mitigation measures. A formal risk-management plan and associated process are used to evaluate each system change request based on assigned risk factors, and to numerically aggregate individual change risks into an overall release risk factor. If a high risk is identified, formal risk-mitigation steps are defined with accompanying manager roles/responsibilities to ensure mitigation measures are executed and tracked.

From strictly a project *process* point of view, maintaining the right balance between high quality and schedule/productivity expectations of customers is a challenge. "As a general rule, customers do not like waiting for requested software changes. DoD customers are no exception," says DCPS System Manager Sandy Barrineau. For a pay system like DCPS, even a perceived *simple* change must be thoroughly analyzed and tested to minimize risk. "From a *people* point of view, the fact that customers' minimum expectations from the system are to be paid (a) correctly and (b) on time – without fail – has been a never-ending challenge," says Barrineau. "DCPS operates 24x7x365, and as such requires a project staff that provides 24x7x365 support."

"Strong program leadership supported by senior management, a strong commitment to success at all levels within the project, and the technical expertise and unceasing determination of the DCPS project staff to provide consistently high quality products and services to our DoD customers are definitely the strongest points of the project," says Meece.

Accomplishments

Since initiating its software process improvement program, DCPS software releases have consistently occurred on schedule and in accordance with the release date committed to the customer. CMM Levels 2 and 3 were achieved while the DCPS project was in the midst of its DoD consolidation effort. In April of 2002, a DFAS site was evaluated using the Malcolm Baldrige National Quality Award Criteria for Performance Excellence and found that "…(DCPS) has effective, systematic processes to ensure security, reliability, and integrity of the systems it develops and maintains."

"This project prides itself in trying to meet every possible need required of its customers," says DCPS SMO representative Michael Johnson. "The role and functions performed by this project are extremely important in maintaining employee morale within the DoD to ensure employees receive accurate and timely payments for the services they perform for the department. The constant and extraordinary efforts expended by this project and organization over the years, coupled with their resilience and desire to deliver quality software projects is to be commended."

"From the initial meeting, the software engineering organization for DCPS proved their professionalism and dedication by outlining the entire conversion process from start to finish to the finest detail," reiterates the director of the only non-DFAS payroll office. "Thanks to their outstanding support and planning, our conversion took place with little to no impact on our office. Whether it's adding additional functionality or incorporating legislative changes, we know it will be incorporated timely and accurately. DCPS has an outstanding configuration control process where customers participate in determining what changes are important and should be incorporated. Another important process is the Software Acceptance Testing, which includes users and policy experts testing the system *before* the software is released. Excellent communication, knowledgeable subject matter experts, and great customer service practices are just a few of the reasons DCPS continues to grow and flourish. They are extremely deserving of this award."◆

Tactical Data Radio System Enhances Combat Effectiveness

Pamela Bowers
CrossTalk

The Enhanced Position Location Reporting System (EPLRS) is a tactical data radio employed by the U.S. Army, Marine Corp, Air Force, and Navy to move key information quickly and efficiently on the battlefield. The EPLRS's layered system software controls all of the communication networks and provides routing services between these networks. The software also performs position location and tracking for EPLRS radios within each network.

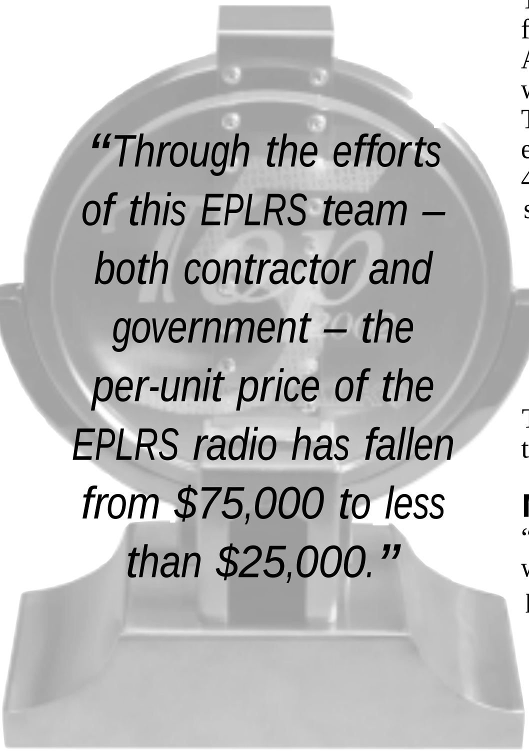
The emerging digitized battlefield holds exciting potential for greater operational flexibility to meet evolving tactical objectives. Among innovations on this front is the Enhanced Position Location Reporting System (EPLRS), the ground tactical data radio system employed in the U.S. Army's Tactical Internet, the U.S. Marine Corps' Tactical Data Network, the U.S. Navy's Amphibious Assault Operations, and the U.S. Air Force's Situational Awareness Data Link.

The EPLRS provides a communications backbone for situational awareness, command and control, and other digital messaging. It consists of a dedicated network of radios that move war-fighting information quickly and automatically. The EPLRS was developed at Raytheon Company's Networked Communication Systems (NWCS) in Fullerton, Calif., using a spiral development approach. At the heart of the EPLRS is its system software, comprised of C-based radio software and Java-based network management software.

In 1999, Raytheon initiated a research and development effort to convert the legacy software and custom operating system (OS) to an Internet-standard layered baseline using a commercial off-the-shelf (COTS) real time OS. "The challenge was to try to make this software, which was running on a hardware platform geared specifically for wireless communications, to run on a generic platform with growth potential," says Michael Born, EPLRS lead systems engineer.

"The software had to support new wireless communication protocols," says Amy Liu, EPLRS system and software project manager. "We had to re-architect the entire radio software and still meet the functionality and performance of the legacy software," says Liu. "This was basically like starting a new project, including documenting and verifying requirements and scheduling to meet completion dates. We had to have a good handle on risks and risk communication."

According to the Raytheon team, the layered architecture facilitates both the use of COTS software and the incorporation of existing software products, such as Raytheon's STAR Router. This ensures interoperability with COTS systems that interface with the EPLRS network, and eases the porting of EPLRS software to other operating systems and hardware platforms.



"Through the efforts of this EPLRS team – both contractor and government – the per-unit price of the EPLRS radio has fallen from $75,000 to less than $25,000."

"EPLRS benefits from mature size estimation processes and historical data for growth profiles," says Sally Cheung, software technical director for the site Engineering Process Group. "We have the benefit at Fullerton of a solid process infrastructure to draw upon. With that infrastructure, we have baseline data capabilities to compare EPLRS performance and product quality."

Software size estimates are made for all major software work activities. Estimates of new, modified, and reused source lines of code (SLOCs) are based on historical data collected from past projects with similar functionality and knowledge of the language being used. These estimates are then verified using the SEER-SEM Software Estimation Tool. Actual counts are collected per build from the output of a SLOC counter. Analysis of size trends in conjunction with other metrics is performed monthly. The total EPLRS software growth did not exceed the established plan by more than 4 percent during the 21-month EPLRS software development cycle.

To meet the customer's need, the EPLRS radio software is required to run on multiple hardware platforms with different operating systems. These platforms include the existing family of radios, future radios (Micro-Light, Joint Tactical Radio Systems) and modeling/training platforms.

Measured Project Value

"Raytheon enabled the radio layered software baseline to be easily ported to multiple platforms, which more than offsets the cost for productization" says Bohdan Kowaluk, senior system engineer at the Tactical Radio Communications Systems Project Management Office, Ft. Monmouth, N.J. The layered software baseline has also enabled new functionality to be added/changed for less than a third of the cost compared to the previous versions of the software.

The continued use and implementation of value engineering changes have reduced cost and made it possible to keep the EPLRS radio hardware design current with respect to the wireless communications industry. Through this process, significant system performance enhancements have been realized, particularly in the areas of data rate and communication services. The latest version is a fully software programmable radio. This enables

The EPLRS team at Raytheon's Fullerton, Calif., site is part of a Capability Maturity Model Level 5 organization.

lower production and logistics costs by allowing future upgrades without requiring new hardware. "The value engineering process has saved the government more than $25.8 million dollars," says Kowaluk.

"Through the efforts of this EPLRS team – both contractor and government – the per-unit price of the EPLRS radio has fallen from $75,000 to less than $25,000," he says. At the same time the capabilities of the system have continued to grow, leveraging the use of COTS hardware and software. As a result of the layered software and the software-programmable hardware, the cost of future upgrades will be a fraction of previous upgrades.

Raytheon Processes Bring Success

The EPLRS program is part of a Capability Maturity Model® (CMM®) Level 5 organization with mature software processes. "There are built-in standard software processes we perform as a CMM Level 5 organization to assure success on all programs," says Cheung. "But on top of that, we apply innovation from a technology standpoint to assess where the product needs to go, and from a process standpoint to assess what innovation must be in place to make things happen."

In addition, the EPLRS program uses the Team of Four (ToF) concept, which is a teamed approach to project leadership, process control and deployment, and metrics analysis. "The Team of Four looks over project performance to suggest improvements and to improve the quality of processes," says Cindy Ruhlman, assistant system and software project manager.

The ToF consists of key stakeholders: the software project manager, the engineering functional manager, the software process engineer, and the software quality engineer. Program management sponsors the ToF and subject matter experts provide support as needed. The ToF meets monthly to analyze metrics, helping to improve process and product quality. Lessons learned are collected, reviewed, and incorporated in future development and test activities.

The EPLRS program achieves delivery of a quality product through configuration control and extensive software testing, says Born. The EPLRS program provides on-site engineering support at customer field locations to ensure a smooth transition when a new software product is released. Field engineers channel feedback to the software team, logistics specialists, depot services, and manufacturing. They generate Test Incident Forms (TIFs) based on user inputs and operational observations as required. Field advisory reports are written to ensure that updated information is disseminated to the field in a timely fashion.

"By using the ToF to manage proactively," says Ruhlman, "analysis showed the TIF cycle time was taking longer than the team would like to resolve problems; improving the TIF process could reduce cycle time." This entailed automating the on-going status of software builds and testing activities. Three separate reporting databases were made relational to automate report generation, streamline TIF review meetings, and enable users to track TIF status on-line, from approval to software build release to test completion. Cycle time measurement has shown a 366 percent improvement with the more efficient process, down from 120 hours to about 30 hours," she says.

The EPLRS team also recognized the opportunity to increase versatility, portability, and maintainability by implementing a Raytheon Six Sigma team, a corporate initiative focused on process simplification and product reproducibility. "It's an opportunity for us to look at areas for improvement. The product, process, the whole thing," says Liu.

Emerging technology is continually monitored for insertion into the product, resulting in design simplification and reduced cost to the customer. "Every time the government procures more radios, we've been able to insert technology and make it simpler and a lower cost with more performance," says Born. "This is possible through a close working relationship with the customer, a long work history, and trust," he says.

Summary

As a CMM Level 5 organization, the Raytheon EPLRS program utilizes proven processes, which quantitatively controls process and product quality through carefully monitored metrics. The EPLRS benefits from continuous process improvements, from taking actions to prevent recurrence of defects, and an effective software estimation process.

In addition the company benefits from an experienced and highly motivated team, says Ruhlman, "This is key to our success."

"We are structured as an integrated process team, which allows the cross communication critical for achieving success," says Born. With this, the use of mature processes, risk mitigation, and the organization's metrics, we are able to achieve a high level of customer satisfaction and deliver a quality product, on budget and on schedule, he says.◆

The JHMCS Operational Flight Program Is Usable on Three Tactical Aircraft

Chelene Fortier-Lozancich
CrossTalk

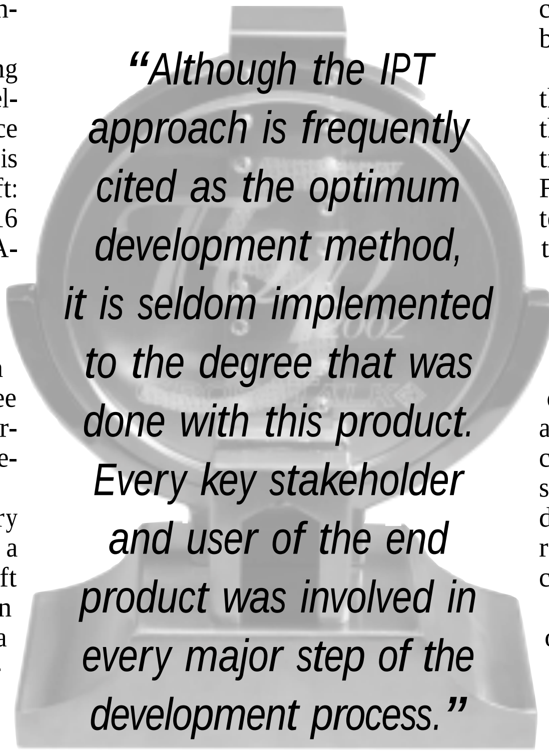
To keep a complex system on cost, on schedule, and at zero delivered defects is no small task. But to do it with one product that can be applied to three different platforms? That was the goal achieved by the Joint Helmet Mounted Cueing System, a high-tech pilot helmet that displays aircraft performance and weapons data on the pilot's visor. It is designed for use by three tactical aircraft, each employing different display formats.

Imagine one product that works across three platforms, contains zero defects, and was developed by several integrated organizations. Got it? Now, add to that the product's ability to enhance performance by way of intuitive training and ease of use and the possibility of future use in fields such as medicine and entertainment. And that's just for starters.

The Joint Helmet Mounted Cueing System (JHMCS) is a high-tech pilot helmet that displays aircraft performance and weapons data on the pilot's visor. It is designed for use by three tactical aircraft: the U.S. Air Force's F-15 Eagle and F-16 Fighting Falcon, and the U.S. Navy's F/A-18 Hornet. Each of these platforms employs different display formats. The goal of the JHMCS Operational Flight Program (OFP) was to be a common system – one product usable by three platforms that still provides for each aircrafts' unique weapons and sensor integration needs.

The JHMCS adds a revolutionary capability to tactical aircraft: Previously, a pilot had to maneuver 25 tons of aircraft in order to bring weapons to bear on an air or ground target. With the JHMCS, a pilot can aim weapons and sensors simply by turning his/her head, completely eliminating the need to point the entire aircraft. The pilot receives feedback – displayed on the visor – of sensors and weapons, including tracked targets, weapons launch and cueing parameters, and aircraft performance information such as airspeed and altitude. Key aircraft performance and tactical data are continuously available regardless of where the pilot looks, providing a significant increase in situational awareness; pilots do not need to divert their attention back to the cockpit displays to obtain needed data – it is always available wherever they look.

That the organizations working on the JHMCS were spread across countries, states, and time zones coupled with creating one product that had zero defects and was rated as highly effective by both F-15 and F/A-18 pilots are but a few of the many reasons that the JHMCS project was chosen as one of the 2002 U.S. Government's Top 5 Quality Software Projects.



"Although the IPT approach is frequently cited as the optimum development method, it is seldom implemented to the degree that was done with this product. Every key stakeholder and user of the end product was involved in every major step of the development process."

Project Need and History

No other system is quite like the JHMCS, according to Boeing/JHMCS Deputy Program Manager Phil King. The closest helmet-mounted system is in the Army's Apache Helicopter. But the applications of the two are very different: the Apache is for low-altitude, heavy terrain nighttime flying. The JHMCS is specifically designed for high-performance tactical aircraft.

"It was originally conceived as a means of putting a heads-up display on the pilot's head," says the Air Force's JHMCS Chief Engineer Patrick Grebinski. "The most driving need for developing the JHMCS was the need to visually aim the AIM-9X short-range high off-boresight missile during close-in 'dogfight' maneuvers," says Grebinski. "The AIM-9X provides a significant increase in air-to-air capability, and a helmet display like JHMCS was the only way the pilots could take full advantage of all the capability provided by this weapon."

OFP-6.1 is the operating software for the JHMCS and is a software upgrade to the previous version, OFP-5.0. At the time of the software project, the F-15 and F/A-18 had concluded developmental testing and were midway through operational testing; the F-16 was just beginning its integration development. As a result, OFP-6.1 had two objectives: to resolve potential F-16 integration issues, and to correct problems found during F-15 and F/A-18 developmental and operational testing, while maintaining compatibility with OFP-5.0 flight-tested software. "Corrections for 20 anomalies, deficiencies, and improvements were required in OFP-6.1," says U.S. Air Force customer Lt. Col. Alton J. Scott.

The OFP was delivered in the form of quality assurance-stamped discs, along with a Version Description Document, which documented the changes in OFP-6.1. Interface Change Requests were used during development to document all changes to the JHMCS aircraft interface. These are currently being incorporated into the JHMCS Interface Control Document, a body of text that defines all pertinent interfaces to the subsystem, including the digital signals passed between the JHMCS unit and the host platform.

OFP-6.1 contains 44.5 thousand (K) core lines of code (CLOC), 23.5K CLOC to communicate with the line-of-sight module, and 20K CLOC to communicate with the symbol generator. The OFP resides in the JHMCS electronics unit, which is a remote terminal on the Mil-Std 1553 aircraft avionics multiplex bus.

HMD = HUD on your Head
307
452
8coo
F-15 HMD
Turn Your Head Instead of the Jet
JHMCS
JOINT HELMET MOUNTED CUEING SYSTEM

The Joint Helmet Mounted Cueing System Display Unit

Approach to Development

"What is … equally noteworthy is the IPT [Integrated Product Team] approach to development of this software," says Scott. "Although the IPT approach is frequently cited as the optimum development environment, it is seldom implemented to the degree that was done with this product. Every key stakeholder and user of the end product was involved in every major step of the development process."

The IPT consists of government and contractor technical experts from each of the three aircraft, plus the Joint Program Office, Boeing (JHMCS contractor), Vision Systems International (JHMCS developer), and Elbit Systems Ltd of Israel (JHMCS software developer). The IPT managed the development of the JHMCS electronics unit OFP. All IPT members attended and participated in requirements definition, design reviews, and testing of pre-release OFP versions. During development, the F-15 and F/A-18 were the leading platforms; the F-16 trailed the other two by a year. The timing was such that F-15 and F/A-18 Initial Operational Test and Evaluation was being completed as F-16 integration was beginning. Initial testing was completed utilizing OFP-5.0 software. Both versions were initially developed by Elbit, which is certified at Software Engineering Institute Capability Maturity Model Level 3[1]. A systems requirements review was held at the project's outset to capture requirements from all three platforms (F-15, F-16, and F/A-18).

Generating budget estimates and prioritizing individual *wish-list* requirements helped establish an affordable list of project requirements, according to King. Regularly held joint working sessions helped resolve issues and disputes. The IPT participated in all design reviews to ensure that needs were being met. A pre-release version was issued to the three platforms for testing in their facilities. This step enabled any errors to be identified and corrected before final release. During this phase, two errors were found and corrected before final delivery.

The end user environment is an operational tactical aircraft and contains its own internal operating system. The OFP processes commands from the host platform mission computer, computes helmet line-of-sight, and generates and renders all display formats. All processing is performed in real-time due to the subsystem's in-flight mission application and human factors involved.

Standard commercial software development tools were used during development, including Emacs, GNU, VxWorks 5.3, and Tornado 1.01 for coding, compiling, and debugging, and MS Office tools for documentation, metrics, design items, etc. Testing, including final verification and validation, was performed in Boeing's Avionics Integration Center, which contains special labs designed to trap, filter, and record various input/output signals and ensure proper integration with the end-user weapons systems.

The JHMCS program met the simultaneous needs of all three aircraft. Because the F-15, F-16, and F/A-18 are also sold internationally, a growing number of international users further complicated the schedules of each aircraft program. Despite these demands, the development process ensured rigid adherence to cost, schedule, and quality requirements through the IPT process, engineering testing, and pre-release testing.

Meeting and Exceeding Expectations

OFP-6.1 is the final engineering event of the JHMCS and had to be kept on cost (to avoid an overrun), on schedule (to meet program need requirements), and developed within existing budget resources. It was delivered one month past the original date to resolve a requirements issue among the platforms that occurred after critical design review. Once final requirements were established, the schedule was revised and all remaining goals met the new schedule. The final version passed the final series of tests and OFP-6.1 was delivered to the three end-user platforms.

"To meet the diverse requirements of three platforms with a single OFP is itself a major accomplishment," says Scott. "To do so and remain on schedule – the only delay was due to a customer requirements issue – and to deliver software with no deficiencies may be an exceptional event in the history of special purpose, real-time, tactical equipment software of this level of complexity." To keep such a complex and challenging system like the JHMCS on schedule, under cost, and at zero delivered defects, while being able to meet the many requirements and demands is a significant accomplishment. It is why the JHMCS is a Top 5 award winner.◆

Note

1. The Elbit software team that developed JHMCS was one of four groups specifically reviewed by SEI when ESL achieved Level 3 certification.

Project Point of Contact

U.S. Air Force
Lt. Col. Alton J. Scott
SPO Director
JHMCS Joint Program Office
Wright-Patterson AFB, OH
Phone: (937) 255-7963

Mike Rietz
Boeing JHMCS Program Manager
Boeing Integrated Defense Systems
St. Louis, MO
Phone: (314) 232-4363

Kwajalein Modernization and Remoting Project Replaces Four Unique Radar Systems With One Common Design

Pamela Bowers
CrossTalk

The Kwajalein Modernization and Remoting (KMAR) project is a five-year program designed to reduce cost and improve capability at the Reagan Test Site by modernizing software and hardware. The challenge was to replace aging, complex, one-of-a-kind radar software systems at each of the four radar sites with a single common design based on Radar Open System Architecture. This article details how the team succeeded in improving the KMAR radar sites while lowering operating costs.

The Ronald Reagan Ballistic Missile Defense Test Site, 2,200 miles southwest of Hawaii on a ring of tiny coral islands called the Kwajalein Atoll in the Republic of Marshall Islands, is home to four unique signature radar sites: ARPA (Advanced Research Projects Agency) Lincoln C-Band Observables Radar (ALCOR), Millimeter Wave Radar (MMW), ARPA Long Range Tracking and Instrumentation Radar (ALTAIR), and Target Resolution and Discrimination Experiment (TRADEX). They are world-class instrumentation sensors that operate at frequencies between 158 MHz and 35 GHz. Due to their age and layered upgrades during the past 35 years, the sensors were badly in need of modernization.

It was very expensive for the U.S. Army to support these vital resources primarily because many highly specialized engineers commuted daily by air from Kwajalein Island to maintain the radars. The Kwajalein Modernization and Remoting (KMAR) project was a five-year program designed to reduce the cost and improve the capability of these radar sites by modernizing all software and hardware except antennas and transmitters.

The challenge was to replace aging, complex, one-of-a-kind radar software systems at each of the four radar sites with a single common design based on Radar Open System Architecture (ROSA). During a 35-year span, the legacy systems had been written in more than a dozen computer languages (primarily Fortran) representing millions of lines of code, ran under a variety of obsolete operating systems and computers, and were difficult to maintain. The legacy systems were manually controlled and lacked simulation and remote operation capabilities. These shortcomings resulted in very high support costs, reliability suffered, and the systems were cumbersome to improve.

The KMAR has been highly successful in solving the legacy systems' deficiencies. The radar sites' maintainability is greatly enhanced by employing a single common software solution that reduces lines of code by more than 70 percent and incorporates modern software languages, up-to-date operating systems, and a rigorous configuration control system. Reliability is greatly improved by thorough testing with advanced simulators prior to software release. By employing a single common solution, a given improvement now only needs to be implemented once to benefit all operational systems. The number and skill level of required personnel has been reduced.

"To eliminate the daily commute ... operators now routinely remotely control and diagnose the radar sites from consoles located in the centralized mission control center on Kwajalein Island.This change alone has saved the range more than $5 million annually."

The Delivered Product

The KMAR hardware and software were installed sequentially during two and one-half years. TRADEX, the fourth radar to be modernized, achieved initial operating capability (IOC) in February 2003. The KMAR software is now the operational control system for all of the instrumentation radar sites. The software comprises a real-time radar control program, operator control and display systems, high-fidelity radar and multi-target simulators, playbacks for review of past missions, automated calibrations, and embedded subsystem software.

About 720,000 lines of code comprise the KMAR software. The radar real-time program (RTP) is approximately 300,000 lines of code. A single common software base now supports all operational radar sites, i.e., each radar runs identical, executable code. This feat was achieved by writing software that is hardware-independent and is fully configured by initialization parameter files. These files describe a radar's unique parameters such as frequency, waveforms, and beam-width. All software systems operate with this same one executable philosophy.

Each radar system, except the antenna and transmitter, was designed and assembled at the Massachusetts Institute of Technology Lincoln Laboratory in Lexington, Mass. All systems were delivered on schedule and on budget. After two years of development, the ALCOR radar was delivered to the site for testing and integration. Four weeks after arrival, it tracked its first satellite. The MMW radar was delivered the following year and the first satellite track was taken three weeks later. ALTAIR was delivered at the end of fiscal year 2002. Both ALTAIR and TRADEX tracked satellites only two weeks after installation on Roi-Namur. This rapid integration was largely due to a highly sophisticated simulation capability that allowed rigorous validation.

The KMAR has improved both the efficiency and productivity of the range, says David Villeneuve, technical director for RTS at the U.S. Army Space and Missile Defense Command. Efficiencies include a permanent reduction of greater than 100 work-years of effort per year, mostly from technical staff reductions in the radar area due to matrixing personnel and streamlining maintenance. The radar sites' staffing levels are less than one-half the levels required 10 years ago. Prior to KMAR, approximately 150 radar support people flew to Roi-Namur daily. The KMAR allowed a greater than 50 percent reduction in aircraft seating requirements.

Productivity improvements include more comprehensive target coverage. All radars can now automatically track multiple objects within their beams, something that ALCOR and MMW could not do previously. In addition to ALTAIR's 128 hours per week of space surveillance support for Space Command, now more TRADEX data is provided for the same or lower cost. Space object identification images have increased from 200 annually to 300 for the same cost.

From 75 percent to 80 percent of hardware and software are now common. The modernized C/C++ code is roughly one-third the line count of the legacy software; system complexity is commensurately reduced. KMAR has resulted in a nearly tenfold reduction in the amount of equipment, says Villeneuve, replacing roughly 100 racks of equipment at each radar by a single row of 12 racks.

Automated radar diagnostic tools have simplified maintenance and operations and improved system flexibility, e.g., by sharing operations and maintenance personnel. To eliminate the daily commute to Roi-Namur, operators now routinely remotely control and diagnose the radar sites from consoles located in the centralized mission control center on Kwajalein Island. This change alone has saved the range more than $5 million annually.

"This is the most comprehensive upgrade attempted and completed in Kwajalein's 40 years of operation," says Villeneuve. "Overall the project is saving the U.S. Army Kwajalein Atoll Reagan Test Site, Space and Missile Defense Command, and the range's customers $17 million per year."

The Development and Build

"The team faced some key challenges," says Stephan Rejto, project manager. "Programmatically, because various groups had been operating these one-of-a-kind systems, no one really believed the four different radar sites could operate with a common system. Technically, trying to isolate the key critical path items was challenging," he says. "Projects fail when you don't identify and tackle the risks up front."

Having the right people carry out a project of this magnitude was also critical. "Many of the people working on the project had a lot of experience with Kwajalein sensors," says Bill Jaros, software manager. "We felt we had a very good command of what had to be done." About 30 percent of the software team were seasoned professionals who could be teamed up with others who were less experienced.

One of the key methods of managing this large software development was to enforce and exploit design modularity. Small teams of one to three developed each subsystem and functional module. Interfaces between modules were clearly defined early in the project. The software team included MIT Lincoln Laboratory and Raytheon RSE engineers at both Lexington, Mass., and the Kwajalein Atoll radar site. This participation by end users in the development and testing cycles significantly minimized final integration and acceptance issues.

"A lot of credit goes to Bill and the system engineering team for decomposing the software into bite-sized chunks that could be assigned to individuals," says Rejto. "That allowed everyone to take ownership of a piece of the system."

Incremental software development was used. "We follow the build-a-little, test-a-lot principal that is clearly the model for incremental development," says Rejto. An initial core real-time capability was established early. Incremental changes were added and tested frequently to minimize risk of failure and simplify regression testing. A loosely coupled architecture facilitated integration. Only working modules were moved to the next level of integration. The process depended heavily on high fidelity radar and target simulators developed as part of this project. Daily testing of new builds with automated scripts ensured constant integrity of the baseline code.

"We were big on having the system operational as soon as possible," says Rejto. "The very first build established a working scaffold for subsequent development. After that first scaffold was up, we always had an operational system. We then integrated new modules and tested to ensure continued operability."

"With these builds, we were able to demonstrate increased functionality to the customer as we proceeded," says Jaros.

Before each operational release, the software underwent a comprehensive integration and test period led by the systems group. All features were tested first with simulators at the development facility, followed by full-system testing with live satellite tracks at site. Finally, prior to IOC, each sensor supported an atmospheric re-entry vehicle mission. In addition, the customer performed a formal acceptance test prior to final validation of each radar.

The KMAR signature radar sites in the order they were modernized, counter clockwise from bottom left, ALCOR, MMW, ALTAIR, and TRADEX.

"The simulator was key to the success of this program," says Jaros. "It was built in-house as part of the program and was used for the entire system checkout." It is a very sophisticated, high fidelity, real-time simulator that basically allows the real-time program to cycle just like it would with the real radar. Jaros adds that it is also very important for operator training. "We believe that about 30 percent to 35 percent of the development effort should be devoted to support tools like the simulator to optimize software development."

With common software, once problems are identified the solution is applied to all radar sites, making each succeeding release more error-free. The current change request rate for the operational radar sites is approximately two per month. Dozens of ballistic re-entry events and thousands of spare tracks have been supported by the modernized sensors. The reliability of these radar sites is demonstrated by the fact that they have been 100 percent successful in fulfilling all mission requirements since becoming operational.◆

Project Point of Contact

Stephan Rejto
MIT Lincoln Laboratory
244 Wood St.
Lexington MA 02420-9108
Phone: (781) 981-3568
E-mail: srejto@ll.mit.edu

The OneSAF Testbed Baseline SAF Puts Added Simulation Capabilities Into Users' Hands

Pamela Bowers
CrossTalk

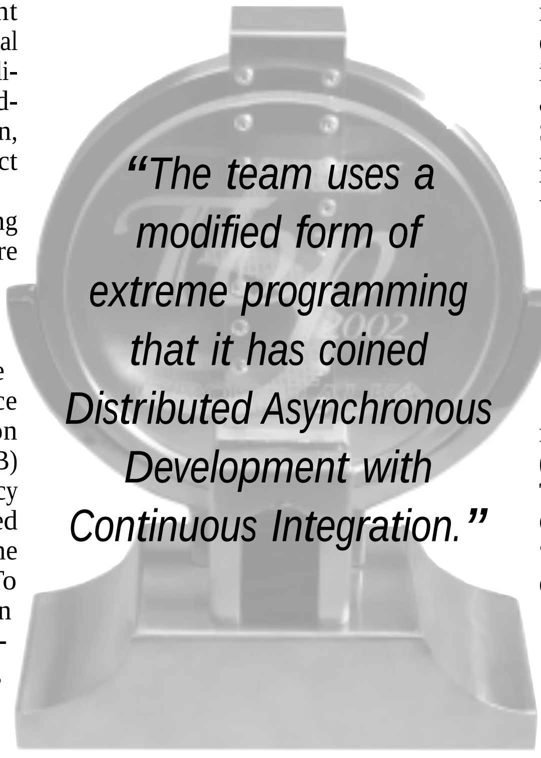
The U.S. Army has used semi-automated forces (SAF) for training, analysis, and research since the mid-1980s. The OneSAF Testbed Baseline SAF (OTB SAF) serves as a bridge between the legacy SAF system and the OneSAF Objective System[1]*. OTB SAF is an open source solution that users can configure to meet their needs.*

Simulation has long been used as a military training tool for land, water, and air combat. The U.S. Army has used semi-automated forces (SAF) for training, analysis, and research since the mid-1980s. These SAF accurately and effectively represent the physical behavior of joint weapons systems as well as the tactical behavior of individual entities and military units. They also portray detailed models of the natural environment (terrain, atmosphere) and the environmental effect on simulated activities and behaviors.

These SAF were in need of updating to allow military training to reflect more modern-day warfare, terrain, and resulting effects on the warfighters, as well as support analysis and research on developing Army platforms. The software program maintained by Science Application's International Corporation (SAIC), OneSAF Testbed Baseline (OTB) SAF serves as a bridge between the legacy SAF system – Modular Semi-Automated Forces (ModSAF) Version 5.0 – and the OneSAF Objective System (OOS). To accomplish this, SAIC developed an open-source solution that maintains configuration management of existing ModSAF capabilities and enhances these capabilities to support interim user requirements. In addition, OTB seeks to reduce risk for OOS development by providing opportunities for integration, test, and user feedback on technology developments.

"The key challenge in this project," says Bryan Cole, SAIC division manager of Simulation and Training Systems, "was to translate the needs of the go-to-war soldier into software requirements that would result in a product that provides the capability the soldiers were looking for, and was sufficiently user friendly."

OTB is used at more than 220 U.S. sites and several international locations. It can be used as a stand-alone simulation, or as an embedded system within a manned simulator. It can also interact in a joint exercise with other live, virtual, and constructive simulations using the Distributed Interactive Simulation and/or High Level Architecture (HLA) simulation standard.



"The team uses a modified form of extreme programming that it has coined Distributed Asynchronous Development with Continuous Integration."

The OTB empowers trainers, analysts, and researchers to configure the simulation to meet their needs without total reliance on software developers. Each version of the OTB puts more and more power into the hands of the users, allowing them to tailor the application for specific requirements.

OTB operates as a distributed system, though the current architecture supports interface to servers (e.g. weapons effects) if desired. Typically, there is no real client or server in the architecture. Workstations negotiate load balancing, and the distributed nature of the application allows recovery from individual system crashes without interruption to the simulation scenario in progress. Methods also exist to participate in a simulation using a distributed network architecture. OTB is easily configured using simple text files and can be modified in the field without needing to be re-compiled from the source code.

OTB 1.0 represents a major overhaul of ModSAF 5.0 code, including the removal of non-functioning libraries, the enhancement of outdated algorithms, implementation of a native HLA interface and the implementation of major new SAF functionality. The update impacted nearly all of the existing ModSAF 5.0 libraries.

Development Method

The Orlando, Fla.-based SAIC group is a Software Engineering Institute Capability Maturity Model Integration (CMMI®) Level 4 organization. This accomplishment and the quality of team members and a progressive customer (Program Executive Office – Simulation, Training and Instrumentation) are what Cole attributes to the project's success. "They have years of disciplined experience applying process to the software development undertaking," he says. "We use basic CMMI processes. Given the nature of the application, it's very complex. However, since its open nature provides a fair amount of latitude to work within the specified requirements to come up with a solution, we encourage creativity and creative thought."

The primary development platform for OTB is a Linux workstation using Debian Linux Version 2.2. A typical hardware configuration is a 1GHz Pentium III with 512 MB RAM and a 60 gig hard drive. Efforts are nearly complete to allow migration of the development platform to Red Hat Linux Version 8.0.

Due to the large installed base of ModSAF users, OTB dictated support of a wide number of operating systems. These include Silicon Graphics IRIX 5.3, 6.2, and 6.5; Sun Solaris 2.5.1 and 2.6; current Debian; Red Hat Linux; and Windows NT. The development team conducts specific tests of new code on

® CMMI is registered in the U.S. Patent and Trademark Office.

each operating system. OTB runs on a wide variety of workstation hardware. Using a minimally configured Linux workstation based on commercially available PC architecture (e.g., 900MHz PIII, 256MB, 4GB HD), OTB can simulate 200 or more individual combat entities per workstation.

The team used spiral or incremental development or a hybrid of the two at various times depending on the effort, says Cole. Multiple developers are on different time lines at any given time depending on the requirements of the particular task, and when it will ultimately be integrated into the baseline. Commercial off-the-shelf products comprise the visual system.

OTB currently consists of more than 1.3 million lines of code in the entire baseline, excluding comments[2]. The source code includes 592 individual libraries. The compiled binary and reader files can be installed as a minimal configuration in as little as 30MB, but the source code tree is more than 135MB prior to compilation.

The team uses a modified form of extreme programming that it has coined Distributed Asynchronous Development with Continuous Integration, says Cole. "The team works in pairs and follows a *build a little, test a little* pattern of incremental development. There are multiple developers addressing requirements at different times during different time lines, requiring support for continuous integration," he says. "This works well at the end because you don't have this big-bang integration."

Long time programmers are typically paired with inexperienced programmers unless there is a specific reason to do otherwise. Recently, Cole says that actual go-to-war operators sat down with SAIC programmers to provide over-the-shoulder feedback. "These guys were the subject matter experts. It's fairly well acknowledged that you get the best product when you have the user right there in the development environment with you. That certainly was true in this case."

The team prides itself on its ability to work in that integrated product team environment. "The concept of a collocated team works," Cole says. The customer, end user, and developer sit down and understand the requirements. Then that understanding is used to develop an incremental build with the user standing over the team's shoulders during testing. "Some of the bedrock of our success is that integrated relationship with the customer, end user, and developers all working according to a common understanding."

The Science Application's International Corporation's OneSAF team gave users configuration power.

Quality In, Bugs Out

As part of its CMMI Level 4 processes, SAIC does an extensive amount of internal peer reviews. Developers use Concurrent Versioning System software to actively track and control code integration and release configurations. All coding changes are peer reviewed, checked by developers, submitted to a single point integrator for inclusion in the developmental tree, and submitted to a separate test group for regression testing and functionality verification.

Once the development team is satisfied, the build goes to the customer or user. "In this way, we may see some changes during the design phase, but we're no longer seeing design changes while we're testing," says Cole. "We get the design stable during the design phase through peer and customer reviews. Once we have the design articulated to the customer, and they acknowledge it will satisfy their requirements, we don't see any changes typically after that point. So we spend a fair amount of time getting that out of the way on the front end."

Cole credits being a Level 4 organization with this accomplishment. "Metrics that we collect concur that we're seeing fewer and fewer errors downstream in the processes as a result of the focus on the front end, in the analysis and design stage, and with peer review. As everybody knows that makes for less expensive software and more predictable schedules." SAIC has experienced a notable reduction in production costs and consistently satisfies a diverse customer base, ranging from development of a Comanche helicopter model to the support of a large scale exercise at Ft. Knox.

Government and contractor managers conduct regular status checks of project progress from both engineering and programmatic perspectives. These reviews are conducted in an Integrated Product Team environment, and are reinforced by routine contractor program management and quality assurance assessments.

"We pride ourselves on on-time delivery of a fully functional baseline," says Cole. "We manage the development process very, very closely using an earned value system so we actually plan what we'll accomplish for each build so we can measure that value and progress as we go. If it looks like something is slipping, we can add additional resources to get caught back up before the ultimate delivery date."◆

Notes

1. An objective system is the term applied to a system that has existence or authority, apart from any individual's experience or thought; in other words, the OneSAF Objective System will stand because of its technical merit.
2. This count was generated using a customized script that SAIC engineers built to automate software-line-of-code (SLOC) counts for the OTB software. This tool counts a full or partial line of executable code as a SLOC, according to industry standard criteria.

Project Point of Contact

Bryan Cole
Science Application's
International Corporation
Division Manger of Simulation
and Training Systems
Phone: (407) 243-3346
E-mail: bryan.a.cole@saic.com

CrossTalk Honors the 2002 Top 5 Quality Software Projects Finalists

Pamela Bowers
CrossTalk

It was difficult to narrow the field from the many successful government projects entered in the second annual U.S. Government's Top 5 Quality Software Projects contest. As a result, the following nine projects are being honored as 2002 Top 5 Finalists. Look for a more detailed article on many of these projects in upcoming CrossTalk *issues.*

AutoREAD

Customer: V-2 Division Onboard the USS Harry Truman CVN-75

The AutoREAD Pilot System addresses the operation of aircraft safety-critical Aircraft Launch and Recovery Equipment (ALRE) onboard U.S. Naval aircraft carriers. It provides a streamlined process for data collection, analysis, and reporting of measurement data not currently available. Flight-critical preventative maintenance tasks are scheduled and downloaded to Pocket PCs. Sailors collect readings using Pocket PCs that automatically perform calculations, tolerance, and range checking against the MRC values identifying marginal and out-of-tolerance readings. Technicians and quality assurance personnel electronically sign off on maintenance actions. The PDA is then re-cradled at the supervisor's branch office where data is automatically uploaded, assembled, and formatted by the workstation application and uploaded to the AutoREAD Server Database for query, reports, and approvals processing. It also facilitates automated trend analysis of ALRE component wear through the up-line reporting of the data from the ship to the in-service engineering support activity at shore sites.

Center Operations On-Line

Customer: 412th Test Wing/ Information Technology Branch, 412th Operations Group

Center Ops Online (COOL) is a web-based enterprise application <https://cool.edwards.af.mil> that provides a secure means to perform mission-critical flight operations tasks. COOL is developed and maintained at Edwards Air Force Base (AFB) and is used by aircrew at Edwards, Eglin, Holloman, and Kirtland AFBs. Users can access COOL from any authorized domain computer, worldwide. Currently, COOL supports more than 600 users and leads the way toward achieving a paperless operations desk. COOL is programmed to comply with Air Force Material Command (AFMC) regulations. Menus, functional buttons, and data entry options are all designed for easy, intuitive navigation, and are filtered based on user authorization. COOL is one of the core applications for Common Operations Enterprise, a common flight management tool set under development to maximize test effectiveness and efficiency for AFMC.

Common User Application Software/Data Management Device

Customer: Electronic Systems Center

The National Security Agency developed the Electronic Key Management System (EKMS) to support loading of key and non-key data to mission support equipment, e.g., encryption devices and radios with embedded communications security (COMSEC) modules. Its intent was to provide a stronger national security posture. The Common User Application Software/Data Management Device (CUAS/DMD) software system is an Air Force developed EKMS component designed to support operations for COMSEC accounts and their users located at any EKMS installation. CUAS/DMD was developed to make the warfighter more efficient by simplifying highly complex EKMS workstations at the user level. The project contributed immensely to lowering the man hours and training required. CUAS provides a highly comfortable user interface to perform extremely critical tasks. DMD allows users to easily navigate information security and COMSEC planning and implementation tool sets, reducing exposure to hostile environments.

Information Access Services

Customer: National Imagery and Mapping Agency

The Information Access Services (IAS) program is a part of the National Imagery and Mapping Agency's National System for Geospatial Intelligence, which provides warfighters and the intelligence community with accurate and current imagery and other geospatial intelligence information. The IAS provides three integrated elements: The Discovery and Retrieval Client 2001, Protocol Adapter, and Profile Services. The Discovery and Retrieval Client 2001 is a powerful data access and retrieval tool that operates much as a Web-based search engine for libraries holding imagery and intelligence data. The Protocol Adapter provides access to libraries not compliant with the most recent specification. Profile Services provides a single point of authentication for user access and a single point of storage for user-specific information, enabling data sharing by multiple clients. IAS is deployed at 12 sites and provides information access to thousands of users worldwide.

Integrated Broadcast Service

Customer: Combatant Commands

The Integrated Broadcast Service (IBS) is a seven-year, $53 million project to develop an intelligence information management system to support a global broadcast communications infrastructure. IBS generates increased combat power by networking sensors, decision-makers, and shooters to achieve shared awareness, increased speed of command, higher tempo of operations, greater lethality, increased survivability, and a higher degree of precision intelligence. The Titan IBS Team delivered the Spiral One product, the IBS Initial Capability (IC), in May 2002 and conducted training and demonstrations for five user commands. The IC System was delivered on schedule and under budget. The IBS will bolster the warfighters' ability to execute threat avoidance, targeting, mission execution, and target negation or destruction.

Logistics Module

Customer: U.S. Air Force

The Logistics Module (LOGMOD) is an unclassified, responsive, user-friendly, online system providing the Air Force, major commands, base-level logistics planners, and base-level unit deployment managers with the capability to plan and execute deployment, reception, and re-deployment operations worldwide. LOGMOD is crucial to planning for worldwide deployment of personnel, supplies, and equipment to meet various exercises, contingencies, and wartime taskings. On-average performance gains of 70 percent realized with version 4.1 afforded global planners extra time to effectively manage every aspect of time-critical deployment operations during force projection, including Operations Enduring Freedom and Noble Eagle. The LOGMOD epitomizes the teamwork concept of extreme programming. The LOGMOD uses a client-server architectural environment consisting of 40 servers located throughout the world.

Milstar Air Force Command Post Terminal Block 6 Software

Customer: SMC DET 11/MCL

Milstar Air Force Command Post Terminal (AFCPT) Block 6 Software is a joint-service satellite communications system. It provides secure, survivable, and endurable communications for the president, secretary of defense, chairman of the Joint Chiefs of Staff, and the Unified Combatant Commanders for their strategic and tactical forces, including command and control elements, aircraft, ships, and submarines, and to command and control the Milstar satellite constellation. This critical AFCPT software is used in fixed, mobile, and contingency terminals, and aboard the Air Force E-4B Airborne Command Post and the Navy E-6B Take Charge and Move Out aircraft. It provides the warfighters interoperable communications with other Milstar terminals on shared networks and enables communications over the Milstar constellation and other existing resources. The upgrade was completed in a dual engineering environment, combining the efforts of two teams separated by 2,000 miles working as an integrated unit.

TaskView 3.2

Customer: Ogden Air Logistics Center/LHM

TaskView allows the user to quickly view an Air Tasking Order (ATO) or Air Combat Order (ACO) at various levels from low-level detail to high-level overview. TaskView displays what was a 100-page plus text document in a variety of textual and graphical formats, including tree structures, tabular layouts, formatted fields, and raw text. In literally minutes, an aircrew can parse/sort the ATO/ACO by mission tasking, display the stick route and associated airspace control measures in FalconView, and then convert the stick route into a combat flight planning software route – a task that once took hours. All routing, refueling points, low-level entry corridors, search and rescue areas, protected airspaces, targets, and all air-related support requirements are fully supported and displayed by TaskView.

Virginia Class Ship Control System Project

Customer: Naval Sea System Command PMS 450

The Virginia Class Ship Control System (SCS) is a revolutionary technological improvement over the Navy nuclear ship control systems of previous submarine classes. A fly-by-wire system now controls the previous hydraulic mechanical-based system, which controls the ship's steering and diving performance. It provides significantly increased capabilities, exceptional reliability via software fault tolerance, and reduced manning via automation. The SCS utilizes commercial off-the-shelf electronic components. It also replaces the conventional hard-wired switches and indicators with four large and four small flat panel displays with touch-screen operator interface. These software-driven displays simplify the operator interface, reduce acquisition costs and installation labor, and provide flexibility and cost savings in life-cycle support. The SCS components communicate with each other via three redundant ship control fiber optic data buses, significantly reducing the cabling required.◆

Top 5 Software Projects Scoring Criteria

Reviewers from the Software Technology Support Center (STSC), Hill Air Force Base, Utah, used the following criteria and point system to score all nominations as part of the process to select the 2002 U. S. Government's Top 5 Quality Software Project's finalists. Each nomination was awarded points (up to a maximum value) based on how well the project performed within each category: customer value, performance, technical value, and reviewer's discretion. At least three STSC consultants or engineers scored each nomination with the top one-third of the nominations closely scrutinized by the internal board to select the finalists.

Customer Value – Maximum 40 Points

Problem Reports

- Were responses to the problem reports and questions timely?

Value

- What was the measured value to the customer's mission (return on investment)?

Benefits and Satisfaction

- Is the end product useable?
- Is the customer satisfied with the end result?
- What other benefits were provided to the customer?
- Was the developer collaborative?
- Did the developer listen to the customer?
- Was the developer knowledgeable? Informative? Helpful?
- Was the developer professional in letting the customer know requirements trade-offs?

Performance – Maximum 25 Points

- Did the developer meet the contracted schedule?
- Did the developer meet the contracted budget?
- How many problem reports have been written against the product since system test?
- Is the customer satisfied with the performance?

Technical Value – Maximum 20 Points

- Was the problem challenging? How hard was this project to implement?
- Was the solution innovative? What approach was used to solve the problem? What technical value did they provide to the world?
- Is the project reusable? Can someone else use the end product, portions of the end product, code, process, or the product's technology to solve a future government problem?
- Is the project repeatable? Given a similar problem, could the group repeat this success or were they just lucky this time? (Did they use defined processes, trained people, etc.?)

Reviewer's Discretion – Maximum 15 Points

Use or don't use these points as discretion dictates. Suggested considerations include the following:

- Previous awards. (CMM, ISO 9000, Malcolm Baldrige, etc.)
- Customers. (Will one small organization use this or will it be dispersed worldwide?)
- Do they have measures that can be used for oversight and additional improvements?
- What is the atmosphere/morale of the developing organization?

STC 2003

15th Annual Software Technology Conference Focused on Strategies and Technologies

Using strategies and technologies to bring about improvement in software processes was the focus of the 2003 Software Technology Conference (STC) held recently at the Salt Palace Convention Center in Salt Lake City. From April 28 – May 1, presenters and speakers brought their expertise and experience on the conference's theme, "Strategies and Technologies: Enabling Capability-Based Transformation," to more than 2,500 attendees from around the world. The STC is one of the largest co-sponsored events for U.S. defense-related software technologies, policies, and practices.

The challenge for future software developers was outlined early in the conference during the co-sponsor's panel discussion. Representatives from the U.S. Air Force, Marine Corps, Navy, Army, and the Defense Information Systems Agency (DISA) each said that the future encompassed true, joint, seamless interoperability among all the military forces. The key to this interoperability is quality software, they agreed.

"We want to get to the fight faster," said Lt. Gen. Peter M. Cuviello, director, office of the Army Chief Information Officer G-6. "True joint, seamless interoperability joins all the forces, as we did in Iraq. That was the most detailed, interdependent operation to date." He stressed that the software development process must allow scalability and predictability.

The 15th annual STC featured 167 different exhibitors and 168 speaker presentations, ranging from software systems/architectures, maturity models, common and open systems to information assurance, best practices and more. In addition to the educational and training opportunities at the STC 2003, the conference gave attendees a chance to network at all levels at a variety of planned events throughout the week.

The STC is co-sponsored by the U.S. Army, Marine Corps, Navy, Air Force, DISA, and Utah State University Extension. The co-sponsors have already started planning STC 2004, scheduled for April 19-22, 2004 in Salt Lake City.◆

Photos by Randy Schreifels of the Software Technology Support Center.

(Photo above) Maj. Gen. Scott C. Bergren, Commander, Ogden Air Logistics Center, Hill Air Force Base, Utah, (right) spoke with Utah Governor Mike Leavitt (left) after the opening general session where Leavitt welcomed attendees to the conference.

(Photo below) Members of the co-sponsors panel (from left) Rear Adm. Michael A. Sharp, U.S. Navy; Lt. Gen. Peter M. Cuviello, U.S. Army; Diann L. McCoy, Defense Information Systems Agency; Robert L. Hobart, U.S. Marine Corps; Dawn C. Meyerriecks, Defense Information Systems Agency; and John M. Gilligan, U.S. Air Force.

(Photo above) Attend
ference break to do sc

USAF
XML Security
BestNing
THE CENTECH GROUP, INC.
TRUST • QUALITY • SERVICE • VALUE
Protecting Critical Infrastructure
Securing IT
Supporting Mission Transition
Consolidating and Integrating Operations
Improvi Workfor Product
Training and Se with Better Tec
HE

(Photo above) Lt. Gen. Peter M. Cuviello of the U.S. Army talked with exhibitors at STC as he toured the exhibitors' show floor. (Photo insert in text) Conference attendees asked questions of individual participants on the co-sponsors' panel.

(Photo above) A mid-week Polynesian theme and entertainment added a note of festivity to the exhibitors' show floor. (Photo left) CrossTalk and Shim Enterprise, Inc. teamed up to offer attendees a chance to test their skills on a climbing wall.

*dees take advantage of a con-
ome networking.*

Measurement and Analysis in Capability Maturity Model Integration Models and Software Process Improvement[1]

Dennis R. Goldenson
Software Engineering Institute

Joe Jarzombek
Office of the Under Secretary of Defense (Acquisition, Technology and Logistics)

Terry Rout
Griffith University

The explicit incorporation of measurement and analysis as a distinct process area in the Capability Maturity Model Integration (CMMI®) provides management with the visibility and focus that organizations need to guide the use of measurement in their process improvement efforts, which was missing in previous models. This article reviews the content and rationale behind the new process area and describes how the ideas introduced there are further elaborated and evolved throughout capability maturity model integration models.

Measurement, or the need for it, is pervasive in software and systems engineering. Yet an understanding of how to best use measurement has remained all too uncommon, as has straightforward guidance from the experts.

Even experts have difficulty following the sometimes-implicit threads among the measurement-related concepts in many process improvement models and standards. Exhortations to do measurement, without sufficiently elaborating what to do, have not worked exceptionally well in the past. However, there has been an increasing recognition of the importance of focusing explicitly on measurement and analysis. Certain basic ideas must be introduced early and well.

The Need for an Early Focus

The need to focus on measurement and analysis from the beginning of a process improvement effort has not always been well understood, even by experienced assessors and expert consultants. Yet organizations that have succeeded in putting successful measurement programs in place often say they could have avoided much grief and struggle with rework had they focused on how to implement measurement and analysis correctly in the earlier phases of their process improvement efforts.

Indeed, some organizations have created their own measurement process areas to guide their improvement efforts, become more competitive, and enhance their ability to more quickly achieve a higher level of process maturity. In fact, measurement is so important to the success of software projects that the U.S. Department of Defense requires the following of all major programs:

> … to have a software measurement process to plan and track the software program and to assess and improve the development process and associated software product. [1]

Focusing on measurement can provide much value to both projects and organizations. Measurement enables project managers to answer the following questions:

- Is there really a problem?
- How big is the problem?
- What is the scope of the problem?
- What is causing the problem?
- Are there related problems?
- Can I trust the data?
- What should I expect; what will happen?
- What are my alternatives?
- What is the recommended course of action?
- When can I expect to see results?

> *“Measurement helps provide objective insight into issues and processes, along with the ability to objectively identify and manage risks and provide early detection and resolution of problems.”*

Measurement helps provide objective insight into issues and processes, along with the ability to objectively identify and manage risks and provide early detection and resolution of problems. Measurement also facilitates evidence-based team communication and enables objective planning and estimating and the ability to assess organizational performance in an unbiased and defensible manner. These in turn provide an objective basis for defending and justifying decisions. Finally, measurement provides information that improves decision making in time to affect the business or mission outcome [2].

Doing Measurement Right

Of course, many approaches to software measurement now exist. Several published international standards address software measurement and closely related issues. There also exists voluminous literature in software measurement and metrics, and a rich body of literature in statistics and quantitative methods dating back well over a century.

Negotiating the morass of standards, models, guidebooks, courses, and expert consultants can be a daunting task. Fortunately, though, there is a clearly emerging community of practice that spans both software measurement and process improvement[2]. In fact, the Capability Maturity Model Integration (CMMI®) Measurement and Analysis process area was developed in a collaborative, coordinated fashion with colleagues who also worked concurrently with the Practical Software and Systems Measurement Support Center, and worked on the development of the emerging ISO [International Organization for Standardization] standards on both software measurement and process assessment. People working in the field are also closely coupled with related work in standards and with groups such as the International Function Point Users Group (IFPUG)[3].

Measurement and Analysis in Capability Maturity Model Integration Models

The Measurement and Analysis process area is an important addition to the CMMI. Its scope is much wider and more explicit than the treatment of measure-

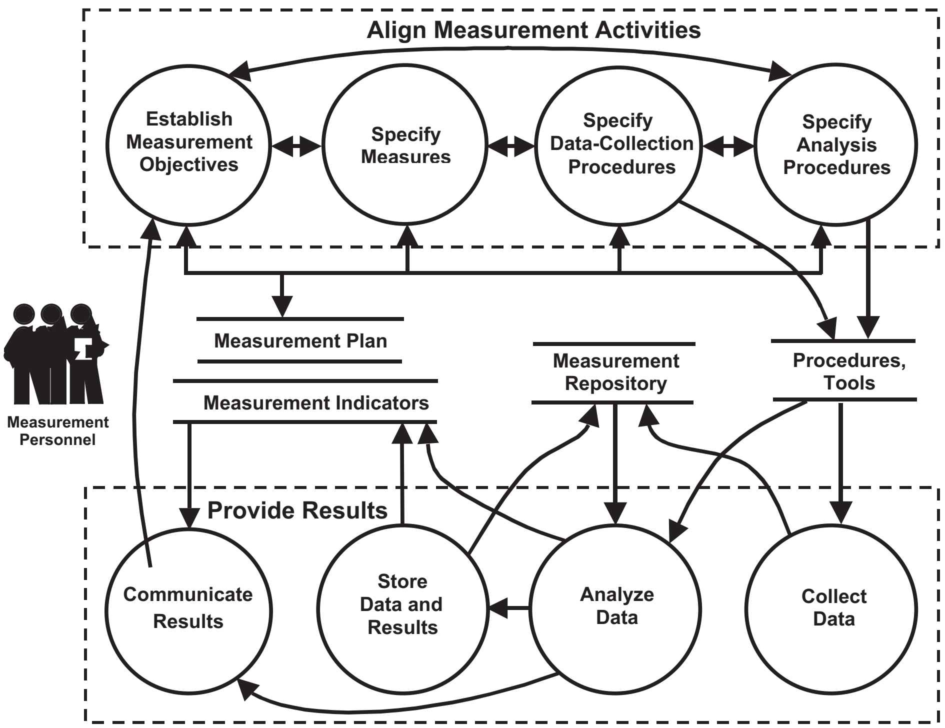


Figure 1: *The Measurement and Analysis Process Area*[5]

ment in the Capability Maturity Model for Software (SW-CMM) [3]. The SW-CMM contains a measurement and analysis common feature, the practices of which apply to the institutionalization of the model's key process areas. Akin to generic practices in capability maturity model integration models, these practices are meant to control and improve the performance of the processes themselves. Some measurement-related practices also exist in various places in the activities-performed common feature in the SW-CMM, but a single, coherent treatment does not exist for what is required to establish and sustain a viable measurement and analysis process.

"The purpose of measurement and analysis is to develop and sustain a measurement capability that is used to support management information needs" [4]. The Measurement and Analysis process area supports all process areas by providing practices that guide projects and organizations in aligning their measurement needs and objectives with a measurement approach that will provide objective results that can be used in making informed decisions and by taking appropriate corrective actions.

As discussed more fully in the process area itself, measurement and analysis practices are organized under two specific goals that are aimed at (1) aligning measurement activities with identified information needs and objectives, and (2) providing data analyses and results that address those needs and objectives. These goals may be achieved by the successful performance of their respective specific practices shown in Figure 1[4].

The specific practices associated with the first goal establish a coherent plan for measurement and analysis. They address these questions: "Why are we measuring?" "What are we going to measure?" "How are we going to measure?" and "What will be done with the data once we have them?" The specific practices associated with the second goal advise the user to just do it. Of course, the ultimate goal is as follows:

> … to get the results of performing measurement and analysis into the hands of those who will take action based on the results. The process area emphasizes the need that results must be communicated to those needing the information. [5]

Other guidance about what constitutes good measurement practices does exist in capability maturity model integration models, most notably in some of the process areas with a legacy in the source documents on which the CMMI is based. Those process areas do contain certain specific practices that require measurement activities to be performed. As seen more fully in the CMMI, the Measurement and Analysis process area makes explicit reference to other process areas – in particular, Organizational Process Definition and the heavily measurement-oriented process areas at CMMI Levels 4 and 5. With the addition of the Measurement and Analysis process area, the CMMI summarizes much of the experience base on which the proper conduct of measurement and analysis relies.

Maturing Measurement Capability

The Measurement and Analysis process area provides a central focus that describes good measurement practice. But the process area does not stand alone. The CMMI also provides important guidance in its generic goals and practices, some of which have explicit measurement content. The generic practices serve together to help institutionalize measurement and analysis, or any other process, and to improve the capability with which measurement and analysis are performed over the life cycle of the product and organization.

Like any other process area, measurement and analysis can progress from being performed in an essentially ad hoc manner, through following a well-defined measurement process, to using measurement to evaluate and improve the measurement process itself. Several CMMI generic practices have a clear measurement flavor (Table 1). However, all of the generic practices can be applied to the conduct of measurement and analysis[6].

Several generic practices discuss organizational policies, sufficiency of resources, explicit assignment of responsibilities, and training provisions. These help estab-

Table 1: *Measurement-Related Generic Practices [4]*

Practice	Statement
2.8 Monitor and Control the Process	Monitor and control the process against the plan for performing the process and take appropriate corrective action.
3.2 Collect Improvement Information	Collect work products, measures, measurement results, and improvement information derived from planning and performing the process to support the future use and improvement of the organization's processes and process assets.
4.1 Establish Quantitative Objectives for the Process	Establish and maintain quantitative objectives for the process about quality and process performance based on customer needs and business objectives.
4.2 Stabilize Sub-Process Performance	Stabilize the performance of one or more sub-processes of the process to determine its ability to achieve the established quantitative quality and process performance objectives.
5.1 Ensure Continuous Process Improvement	Ensure continuous improvement of the process in fulfilling the relevant business goals of the organization.
5.2 Correct Root Causes of Problems	Identify and correct the root causes of defects and other problems in the process.

lish and sustain process capability and a commitment to doing measurement regularly and well. Indeed, they provide the organizational infrastructure that is necessary to implement and institutionalize any process.

Other generic practices provide guidance for planning and related activities. The planning-related generic practices, including the establishment of quantitative objectives (4.1) and improvement objectives (5.1) help establish the scope and objectives for measurement work. Although they relate to both specific goals of the Measurement and Analysis process area, they are particularly important for the alignment activities discussed in specific goal 1 listed earlier.

Several generic practices, including stabilize sub-process performance (4.2), guide the performance and management of any process. These also support both specific goals 1 and 2 of Measurement and Analysis, but are particularly important for doing measurement and analysis and reporting the results.

Finally, along with others, the measurement-oriented generic practices that require monitoring and controlling the process (2.8), collecting improvement information (3.2), and correcting root causes of problems (5.2) help evaluate and improve the conduct of the measurement process itself. Together they provide an evidential basis for improving the manner in which future measurement and analysis are done[7].

Implementing Good Measurement Practice

Along with the measurement-related generic practices, the Measurement and Analysis process area provides essential guidance about what to do whenever there is a need for measurement. However, measurement and analysis are always done in the context of performing other processes.

The Measurement and Analysis process area seamlessly integrates measurement activities with those other processes to address a wide variety of both project and organization-wide information needs and provides a basis for integrating measurement and analysis with process definition.

Like other support functions, the Measurement and Analysis process area serves multiple purposes. Every process area is dependent to some extent on properly using measurement and analysis. The engineering and management process areas describe the sources of the contractual requirements, other information needs, and business objectives with which the measurement and analysis activities are aligned. In turn, the results of the measurement activities are provided back to inform the work described by those same process areas.

Maturing Analytic Capability

Measurement is applied differently as the organization successfully satisfies the goals of more and more CMMI process areas. It typically begins with a focus on clarifying sometimes-implicit business objectives and informational needs and translating them into measurable objectives. A basic set of skills, resources, and experiences is built for the future. Measurement often starts with using simple charts and graphs, but as the organization matures, demand increases for more sophisticated quantitative analyses such as statistical process control (SPC), structural modeling, or other multivariate statistical methods.

> *"Moreover, relying on a well-defined measurement process can demonstrate evidence of business value, thus helping justify continued investment in process improvements and the measurement and analysis activities that support it."*

As the organization's analytic sophistication increases, finer-grained measures of defects and product quality are coupled explicitly with process performance. By the time the organization reaches CMMI Level 4, routine reliance on quantitative management enhances process discipline. After Level 5 is attained, there is increased use of work-product inspections, systematic programs of defect prevention driven by causal analysis, and improved process performance that often leads to markedly increased predictability of productivity and schedule.

Analytic Approaches

Many statistical analytic solutions to measurement problems are possible in software process improvement; however, few are widely used. For example, in an unpublished survey of representatives of high-maturity organizations, almost 90 percent reported that SPC control-charting techniques were in common or standardized use in their organizations [6]. Only Pareto analyses[8] appear to be more common.

Such wide reliance on SPC and Pareto analyses is due in large part to the SW-CMM and its heritage in industrial engineering and total quality management, and also because graphical presentations are intuitive[9]. Of course, not all problems have the same solution. SPC, for example, is only one tool, and it is not always used correctly. One size does not necessarily fit all, yet there still is relatively little evidence supporting use of alternative data analytic approaches.

Although the use of designed experiments and quasi-experimentation fits quite naturally into applications of causal analysis and defect prevention [7, 8], they still are not widely used in higher-maturity organizations; fewer than 10 percent reported that they used designed experiments, and only two respondents said they used quasi-experimental designs [6].

There is evidence, however, that experimental methods may be used more commonly than you might think. For example, in a recent study of practitioners and users of software measurement, almost 40 percent said their organizations commonly employ experiments and/or pilot studies prior to the widespread deployment of major additions or changes to development processes and technologies. Undoubtedly, not all of them follow rigorous methodological standards. However, the results are encouraging because the study sample was structured to include representatives from organizations that have had varying success with their software measurement efforts; there were failures as well as successes [9].

One occasionally sees more sophisticated uses of curve fitting, for example, using Rayleigh curve-based models[10]. Six Sigma approaches also are gaining increased interest in the process improvement community [10, 11]. However, widespread use remains uncommon [6]. Indeed, there still is minimal use of multivariate methods, classical or otherwise. Even basic statistical methods such as regression analysis or analysis of variance are reportedly used by relatively few high maturity organizations [6, 9].

Available Guidance

With the exception of SPC, capability maturity model integration models do not offer a great deal of guidance on using data analytic methods. Indeed, even ISO/IEC 15939 [12] focuses more on measurement fundamentals as opposed to detailed guidance about analysis. With the notable exception of the Practical Software and Systems Measurement Support Center's work and their Practical Software Measurement Insight tool[11], the same is true for all of the best-known frameworks, models, and standards in the process improvement community. Yet any raw data must be analyzed and interpreted with care.

The early emphasis is on presentation graphics in most software measurement programs, probably because visual displays often appear to be intuitive. But they also are often misused and misinterpreted [13, 14]. The challenge to the measurement community is to balance rigor and methodological defensibility with clarity and practical import. There is often a very real culture clash between measurement experts who are trained to attend to excruciatingly obtuse detail and practitioners who need actionable guidance.

Classic tools for process improvement in the manufacturing world date back at least to Deming [15] and Juran [16]. Techniques such as Pareto charts, run charts, histograms, pie charts, scatter diagrams, bar graphs, and control charts are first principles for any good manager or practitioner in most engineering disciplines. These are not advanced topics by any means, although their proper application and interpretation do take training and experience. Indeed, control charts are often posted on the walls for easy reference in many enterprises, but not often in software organizations, which is something of an anomaly. The software process improvement community often seems to have forgotten its heritage in total quality management and industrial engineering.

Detailed, prescriptive, how-to guidance is outside the province of capability maturity model integration models. But other sources of guidance do, of course, exist. Many other books and articles exist in the published literature on both applied statistics and software measurement[12]. And many courses for measurement practitioners also are available.

Summary and Conclusions

A successful process for measurement and analysis is characterized by decision making that regularly includes data analyses results that are based on objective measurement. Following such a process can help projects and organizations make significant performance improvements in their other software processes and in the products and services that those processes help bring about. Moreover, relying on a well-defined measurement process can demonstrate evidence of business value, thus helping justify continued investment in process improvement and the measurement and analysis activities that support it.

The sophisticated use of measurement and analysis that characterizes high-maturity organizations has been shown to result in substantial added value [17]. Organizations that have attained high-maturity status regularly report notable improvements in measured customer satisfaction as well as better schedule and budget predictability. These businesses also commonly provide evidence of increased staff productivity, as measured, for example, by reductions in development effort per line of code or function point. Also they demonstrate heightened product quality with earlier defect detection profiles and marked decreases in defect density during testing.

Measurement is a key enabler for process improvement and enhanced product quality. An organization with a mature approach to measurement and analysis will have confidence in its abilities to effectively deliver products that meet its customer's needs. Measurement must begin early if it is to reach its full potential, and measurement capability must grow over time. It is difficult for us to conceive of serious software engineering or accomplished management without measurement. The time has come for software and systems development to move toward using the same degree of sophistication in measurement that other engineering disciplines have used for many decades.◆

Acknowledgements

We owe a great depth of gratitude to many people too numerous to mention here, but special thanks are due to David Card, Jim Curfman, Khaled El-Emam, Wolf Goethert, Will Hayes, Lauren Heinz, David Herron, Seshadri Iyer, Cheryl Jones, Jim McCurley, Jack McGarry, Mark Paulk, Jerome Pesant, Kevin Richins, Janet Russac, Sandy Shrum, Guy Taylor, David White, and Dave Zubrow.

References

1. Gansler, J. S. "Software Evaluations for ACAT I Programs." The Under Secretary of Defense Memorandum. 26 Oct. 1999.
2. McGarry, J., D. Card, C. Jones, B. Layman, E. Bailey, J. Dean, and F. Hall. Practical Software Measurement: Objective Information for Decision Makers. Boston, MA: Addison-Wesley, 2001.
3. Paulk, M. C., C. A. Weber, B. Curtis, and M. B. Chrissis. The Capability Maturity Model: Guidelines for Improving the Software Process. Reading, MA: Addison-Wesley, 1995.
4. CMMI Product Team. Capability Maturity Model Integration (CMMI®) Version 1.1 Continuous Representation. Pittsburgh, PA: Carnegie Mellon University, Mar. 2002.
5. Zubrow, D. "The Measurement and Analysis Process Area in CMMI." ASQ Software Quality Newsletter 2001.
6. Paulk, M., D. Goldenson, D. White, and M. Zuccher. Unpublished data from a study of high maturity organizations, 2002.
7. Wohlin, C. et al. Experimentation in Software Engineering: An Introduction. Boston, MA: Kluwer Academic Publishers, 2000.
8. Goldenson, D., and R. Stoddard. "The Use of Designed Experiments in Software Engineering." Empirical Studies of Programmers: Sixth Workshop. Eds. W. Gray and D. Boehm. Norwood, NJ: Ablex Publishing Corporation, 1996.
9. Goldenson, D., and K. El-Emam. "What Should You Measure First? Lessons Learned from the Software CMM." Software Engineering Symposium, Washington, D.C., Sept. 2001.
10. Card, D., "Sorting out Six Sigma and the CMM." IEEE Software May/June 2000.
11. Siviy, J. "Six Sigma: An Overview." Pittsburgh, PA: Software Engineering Institute, 1 May 2001 <www.sei.cmu.edu/str/descriptions/sigma6.html>.
12. ISO/IEC 15939:2002. "Software Engineering – Software Measurement Process." 11 July 2002 <www.iso.org>.
13. Tufte, E. The Visual Display of Quantitative Information. Cheshire, CT: Graphics Press, 1983.
14. Goethert, W., D. Goldenson, and W. Hayes. "Scatter, Line, Pie, and Bar: Using Charts to Make a Point." Software Engineering Symposium, Pittsburgh, PA, Sept. 1998.
15. Deming, W. E. Out of the Crisis.

Cambridge, MA: MIT Center for Advanced Engineering Study, 1986.

16. Juran, J. M. Juran on Planning for Quality. New York, NY: MacMillan, 1988.
17. El-Emam, K., and D. Goldenson. "An Empirical Review of Software Process Assessments." Advances in Computers. Ed. M. Zelkowitz. San Diego, CA: Academic Press, 2000.

Notes

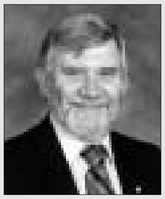
1. This text is an abridgment and extension of material that previously appeared in the following: International Function Point Users Group. IT Measurement: Practical Advice from the Experts. Eds. D. Herron, J. Russac, D. Coley, J. Curfman, B. Emmons, and J. Schofield. New York: Addison-Wesley, 17 Apr. 2002: 577-604.
2. We called it the "Measurement Mafia" when we were creating the Measurement and Analysis Process Area for CMMI.
3. See, for example, Emmons, B., and C. Dekkers. "Function Point (FP) Maturity Model: How FP Supports the Capability Maturity Integration (CMMI) Model." CrossTalk Feb. 2002.
4. The data flow arrows in the figure are modified from the original.
5. This figure is based on a similar one in the CMMI Training Materials, which are not publicly available.
6. The higher capability level generic practices are not included in the staged version of the V1.02 CMMI models.
7. In fact, measurement experts routinely subject their own work to empirical evaluation.
8. For an introduction to Pareto analysis, see Florac, W., and A. Carleton. Measuring the Software Process: Statistical Process Control for Software Process Improvement. Reading, MA: Addison-Wesley, 1999: 62-64
9. Ibid. This is a good source for a useful treatment of SPC in software process improvement.
10. A Rayleigh curve yields a good approximation to the actual labor curves on software projects.
11. The PSM treatment of data analysis focuses on graphical presentations. Their examples range from the very simple to sometimes quite sophisticated uses of multivariate graphical indicators. See McGarry, J., D. Card, C. Jones, B. Layman, E. Bailey, J. Dean, and F. Hall. Practical Software and Systems Measurement: A Foundation for Objective Project Management. V4.0b. Department of Defense and U.S. Army, Oct. 2000, particularly part 4, chapter 3.
12. A few good places to start include Van Solingen, R., and E. Berghout. The Goal/Question/Metric Method. New York: McGraw-Hill, 1999; Fenton, N. E., and S. L. Pfleeger. Software Metrics: A Rigorous and Practical Approach. 2nd ed. Boston, MA: PWS Publishing Company, 1999; and Florac, W., and A. Carleton. Measuring the Software Process: Statistical Process Control for Software Process Improvement. Reading, MA: Addison-Wesley, 1999; as well as the work by Tufte [13], Wohlin [7], and Card [10] cited earlier. Journal articles with examples of data analytic methods worth emulating often can be found in Empirical Software Engineering: An International Journal ["International Symposium on Empirical Software Engineering." IEEE Computer Society Staff, 2002].

About the Authors

Dennis R. Goldenson is a senior member of the technical staff in the Software Engineering Measurement and Analysis group at the Software Engineering Institute. His work focuses on using measurement and analysis in software engineering, the improvement of process appraisal methods and models, and the impact and transition of software process improvement and other software engineering practices. He is a principal author of the "Measurement and Analysis Process Area for CMMI" and served as co-lead of test and evaluation for the project.

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890
Phone: (412) 268-8506
Fax: (412) 268-5758
E-mail: dg@sei.cmu.edu

Joe Jarzombek is director of Software Intensive Systems in the Office of the Under Secretary of Defense (Acquisition, Technology and Logistics). He serves there on loan from the Institute for Defense Analyses in Alexandria, Va. A project management professional, certified by the Project Management Institute, he previously served as vice president of Product and Process Engineering, for the USERTRUST Network, a Public Key Infrastructure that provides security and legal protections for Internet transactions. Prior to his retirement from the military, Jarzombek was a lieutenant colonel in the U.S. Air Force, he was manager of the Air Force's Computer Resources Support Improvement Program Office, and he was sponsor of CrossTalk, *The Journal of Defense Software Engineering*. He also is a principal author of the "Measurement and Analysis Process Area for CMMI."



Terry Rout is a senior lecturer in the School of Computing and Information Technology at Griffith University, Queensland, Australia, and is associated with the Software Quality Institute there. He lectures in the areas of software engineering, software quality, and project management. Rout is chairman of the Australian Committee for Software Engineering Standards, and has been a member of the Australian delegation to the International Committee on Software Engineering Standards since 1992. He is the project editor for ISO/IEC TR 15504-Software Process Assessment. Rout also is a member of the newly established Capability Maturity Model Integration (CMMI®) Expert Group, which advises the Software Engineering Institute on issues related to the CMMI adoption and interpretive guidance.

Combat Resistance to Software Measurement by Targeting Management Expectations

Carol A. Dekkers
Quality Plus Technologies, Inc.

The software industry has been slow to embrace measurement practices even when software managers recognize the benefits it can deliver. This article seeks to overcome management resistance to software measurement by addressing management expectations. It includes a discussion of the human and technical factors involved in software measurement success.

When it comes to measurement, the software industry responds inconsistently. Even though other industries have long depended on measurable outcomes to gauge their profitability and control their processes' progress, the information technology industry has been slow to embrace software measurement. Even when software managers recognize that software measurement can deliver benefits and is a critical component in achieving a Software Engineering Institute Capability Maturity Model for Software (SW-CMM) Level 3 or higher, their expectations are often unrealistic.

While advancements have been made to implement software measurement, specifically to the Department of Defense (DoD) in such initiatives as the Practical Software Measurement Initiative and others, for the most part software measurement is not well understood or used in the software industry. Traditionally, when the software industry has produced metrics, they are typically isolated operational measures (such as transactions per hour or million-instructions per second) or physical measures (such as source lines of code) that tell little about the effectiveness or efficiency of the development process.

Additionally, many software managers seek a silver-bullet metric, which not only answers development questions but does so with several-decimal-point accuracy. Because there is no silver bullet, measurement falls short of these expectations and metrics programs are abandoned before they deliver a return on investment. Such outcomes do not need to happen, in fact, software measurement can deliver value even when the chosen measures are subjective (as with customer satisfaction) or when the measures are imperfect (as with defect tracking).

This article seeks to overcome management resistance to software measurement by addressing management expectations for silver-bullet metrics. Realistic management expectations for measurement provide a chance for measurement to survive. It is a critical prerequisite for DoD agencies contemplating process improvement based on software measurement.

Not an Exact Science

For engineers, computer scientists, and other information technology professionals, it is natural to expect that measurement can be made into an exact science (recall college labs where data outliers on research graphs were too difficult to explain and therefore were erased). In the real world of information technology, however, measurement does not always translate into predictable outcomes, and not everything that can be measured necessarily should be.

Measurement consists of taking a series of observations about a process or product and analyzing the data to indicate where positive changes might be made. It is important to realize that just because something can be measured to the *nth* degree of accuracy does not make it valuable to measure – there needs to be a purpose and a method behind the measure before it will be useful.

> *"... there needs to be a purpose and a method behind the measure before it will be useful."*

The first step in creating a successful measurement program is to realign your and your company's expectations about software measurement. The following sections describe how to do that.

Goal-Question-Metric Approach

Follow the Goal-Question-Metric (GQM) approach to software measurement introduced by Victor Basili of the University of Maryland[1]. This approach forces companies to clearly identify their strategic goals and to pose questions to track whether or not the goals are being met. Only then are the metrics needed to answer the questions identified, and data collection mechanisms put into place. The resulting metrics necessarily depend on the specific goals and questions of the organization. Within the SW-CMM are a number of Level 2 and Level 3 key process areas that can form the basis of an organization's goals/questions/metrics.

The importance of planning when implementing measurement is an area that is often glossed over in an organization's rush to quickly establish a solid metrics program. As such, it is not uncommon for senior management to initiate a software metrics program by collecting metrics without first having identified the goals or the questions into which the metrics should fit. Consequently, after six months of data collection, the lack of planning often becomes clear as management and their developers try to fit the metrics into a cohesive program to support their goal/question/decision-making needs. Without planning, the collected metrics often do not fit together properly, nor do they answer the questions that management needs answered to gauge whether their goals are being met.

Planning for measurement (by identifying your critical goals and questions that measurement must support) is as important a prerequisite as are requirements in software development. To be successful, measurement implementation should follow a project plan and consist of allocated and scheduled resources to perform the measurement requirements (GQM). An analysis of how measurement will be done should address the six W's of data collection and measurement:

- What processes will be impacted?
- What measures are needed?
- Who will participate in metrics design, collection, data analysis, and reporting?
- When will metrics be collected, including collection frequency (how often), life-cycle phase, data entry, etc?
- Where will metrics be collected: centralized/decentralized, from all projects/some projects, etc.?

- Why are metrics being gathered? (The purpose of each metric can change what is reported.)

Metrics without goals and pertinent questions are meaningless. For example, if a manager asks me for my project hours, and I think the purpose is to determine my paycheck, the answer will be 40 hours per week. If the purpose is to analyze how much time is actually being spent on the project by phase so that the process can be improved, my answer will be the accurate 50 hours per week.

While it may be tempting to rush directly to the design and selection of metrics, *do not* skip proper planning! Instead, take the time and energy to develop and produce a measurement project plan. This project plan is critical to aligning management's expectations because it will identify the resources, the time frame, and the coordination needed to implement measurement. In the same way that skipping software requirements leads to products that do not meet customer needs, skipping measurement program requirements – GQM – will lead to a measurement program that does not meet its customer needs.

The aforementioned book[1] outlines more details about the steps to take in planning a GQM-based measurement program, including checklists, tasks, recommended time frames, and resource levels. While it is not the only model for implementing software measurement, (others include the Balanced Scorecard), GQM is a rational approach that aligns metrics to the business goals, which in turn, will lead to higher success rates for measurement programs.

No Silver Bullet

Communicate early and often that there is no silver-bullet software metric, just as there is no silver-bullet accounting metric. Defects, functional size, project duration, and work effort all measure a different aspect of software development; they are not interchangeable. No single measure or single combination metric will satisfy all goals or answer all measurement questions – you must choose the metric suitable for each specific question. Once the specific, measurable GQMs have been identified, select the most appropriate metric. In the same way that a toolbox contains many tools, each specifically designed to serve a particular use, a measurement toolbox should contain specific measures selected to suit your specific needs.

For example, if the goals were to increase user satisfaction and software product quality, the questions would include the following: "What was the level of customer satisfaction with the product before implementing change?" "What is the new level of customer satisfaction?" "How has product quality improved (percentage increase in product quality levels)?" The contributing metrics would then consist of the following: customer satisfaction rating (using a numerically scored customer satisfaction survey) and defect density (measured using defects per function point or other software sizing measure).

There is no Swiss army knife of metrics – you need to select the measure(s) that best fits the purpose, be it defects, function points, number of objects, lines of code, customer satisfaction, work effort, etc. – each is intended to measure a different aspect of software development.

> *"Metrics without goals and pertinent questions are meaningless."*

Learn About Metrics

Learn about the available metrics and what they mean before implementing them in an organization. For example, work effort is a function of many variables, including software size, implementation technology, development tools, skills, hardware platforms, degree of reuse, tasks to be done, and many others. As such, no single variable can accurately predict work effort; yet, there is often an expectation that a single variable (for example, degree of reuse) can accurately predict work effort.

If one of your goals is to increase estimating capability, it is also wise to research the available automated tools on the market and talk to actual users (not just tool vendors) about how their chosen tools work within their particular environment. Note that not all estimating tools address the same problem – some provide probable estimates of work effort and cost, while others provide hourly breakdowns of predicted work effort. Which one will best suit your needs? It depends on your goals and questions.

Use Metrics Properly

Plan a measurement program by using metrics and measures in their intended manner, and ensure that there is a common understanding of the chosen measures. For example, functional size reflects software size based on its functional user requirements, not its physical size. (Physical size of software is often expressed in lines of code.) Together with other variables, functional size can be used as a technology independent measure of software size in order to predict effort or cost in software estimation models.

However, functional size is not the right measure for predicting direct access storage device space requirements. These requirements depend on the physical space taken up by the software and the volume of data and are better measured with other units. For example, 50,000 COBOL lines of code take up more space than the equivalent lines of Java code. And the user requirement to store 50 million transactions takes up more physical space than it does to store a tenth of that.

There is an abundance of information on the Internet about various software metrics from organizations such as the Quality Assurance Institute <www.qaiusa.com>, the American Society for Quality <www.asq.org>, and the International Function Point Users Group <www.ifpug.org>.

Realize True Accuracy

Remember that the accuracy of a metric is a function of the least accurate component measure it involves. People often run into measurement difficulty when they assign several decimal places of accuracy to metrics that are derived from a series of relatively inaccurate or imprecise measures. For example, the function point (FP) count of a project is calculated by summing up discrete values of its component functions, none of which is more granular than three FP. To then calculate defect density and report it with multiple decimal places leads to the mistaken conclusion that the metric is exact.

The same situation arises when sophisticated estimating models produce effort estimates to 15-minute accuracy based on input variables that may have been guesses (e.g., project risk on a one-to-five scale). We all know intuitively that estimates based on a myriad of input variables cannot accurately predict schedules to the closest 15 minutes (let alone the number of hours). Yet I routinely encounter professionals who cite hour estimates with at least one decimal place. (Does this imply that your estimate is accurate to the closest tenth of an hour, or six minutes?)

Correlate With Common Sense

Use common sense and statistics to correlate col-

lected data, and question figures that seem out of line. Do not accept data purely at face value without verifying its consistency or accuracy. Many companies collect work effort data on completed projects, but the definition of project work effort can vary widely across different teams (e.g., overtime recorded/not recorded, resources included, work breakdown structure, commencement/finish points, etc).

Be careful not to compare data that appears comparable because of common units (e.g., hours) that is actually based on different measurement criteria. For example, two projects may report 100 development hours, but one project included overtime and user training hours while the other did not. Although the units are the same, the hours are not comparable. Project hours has no industry-wide definition and can vary widely. Ensure that your organization has established a consistent definition for collecting and reporting project hours for any projects included within the scope of data collection.

Additionally, it is important to apply common sense when establishing the frequency and granularity (unit size) for metrics data collection. For example, while it might be ideal from a theoretical point of view to collect work effort metrics to the closest 0.5 hour broken down by a work breakdown structure task level, it may require more administrative changes and double data entry effort, eliminating potential gain. If your current work effort reporting provides for the developers to enter their project hours into an automated system to the closest hour and broken down by phase on a weekly basis, it would likely prove counterproductive to ask them to re-enter hours a different way just to populate the metrics database. Work with and leverage your existing processes – your developers will appreciate it *and* will more readily buy in to participating in the metrics collection process.

The frequency and granularity of your metrics collection process will depend on your chosen metrics (in support of the goals and questions) and the scope of your measurement program. If your goal is to improve a particular process (e.g., Capability Maturity Model® key processes) for which there has never been any data collected, do not structure the data collection process to impede the overall development processes. Measurement should always be the means to an end – not an end in itself. In other words, measurement must support and provide the opportunity to improve a particular process, *not* to take the place of the development process itself. Measurement should not interfere with the business of developing software. If we focus on measurement to the detriment of developing software, our business will cease to be viable; it will no longer be a matter of measurement, it will become a matter of survival.

Conclusion

These are a few of the factors, both human and technical, that can lead to software measurement success. There is a great deal to be gained by tracking and controlling software development through measurement – if only management would realign their measurement expectations of what the particular measures can provide, rather than seeking a non-existent silver bullet that will solve all of their measurement needs.◆

Note

[1] McGraw-Hill published a book featuring a foreword by Victor Basili: The Goal/Question/Metric Method by Rini van Solingen and Egon Berghout.

About the Author

Carol A. Dekkers is a leading software measurement authority and president of Quality Plus Technologies, Inc., which provides professional software measurement training and consulting services as well as International Function Point Users Group (IFPUG) certified function point training, mentoring, and consulting services. She is past president of the IFPUG and was named by the American Society for Quality as one of the 21 New Faces of Quality for the 21st Century. Additional measurement articles by Dekkers, including how to set up measurement programs, are available by e-mail or by accessing an article request form at <www.qualityplustech.com>.

Quality Plus Technologies, Inc.
8430 Egret Lane
Seminole, FL 33776
Phone: (727) 393-6048
Fax: (727) 393-8732
E-mail: dekkers@qualityplustech.com

Life Cycle of a Silver Bullet

Sarah A. Sheard
Software Productivity Consortium

"Attention! Throw out those other improvement methods – we have just discovered the best ever. With our method, your quality will go up and costs and cycle time will go down." Almost any improvement method is hailed as the best way to save business from problems when it is new. Unfortunately, a few years later, this same method is now the reviled, flawed method that a new method is replacing. This parable tells how this happens.

In the 17th century, Europeans believed that silver bullets could kill werewolves. Today's executives seek silver bullets to protect themselves not from werewolves but from sliding profits, disillusioned stockholders, and lost market share. The silver bullets for our executives are those new management trends that promise to transform the way business is done. Examples over the decades have included Management by Objectives and Total Quality Management, while Six Sigma, Lean Enterprise, the Capability Maturity Model Integration (CMMI®), and agile software development techniques are more recent methods earning silver-bullet reputations. Process improvement initiatives like these can and do work, but how they are implemented is critical to their success. The following parable shows the 11 phases in the life cycle of such an improvement initiative.

Phase 1: Fresh Start

An executive of Porcine Products, Mr. Hamm, decides to throw away all silver bullets. He decides that no one knows his company like he does. He takes a close look at how the company is working to determine what its problems are and how they arose. He also looks at company strengths to leverage them and make them more effective in the future.

Envision a little pig in a suit, wiping a bunch of architectural drawings and books off a table.

Phase 2: Executive Dedication and Openness

Hamm makes it his single-minded focus to improve Porcine Products. Having identified its problems and strengths and determined how to address them, he dedicates time and money to implementing the identified improvements and eliminating conflicting initiatives. He hires forward-thinking, intelligent managers and devotes considerable amounts of his own time to be sure that the problems are truly solved, not just glossed over. Hamm and his managers research a number of current and future improvement methods to help define current problems and to be potential tool kits providing applicable suggestions.

The executive insists that the senior managers become part of the solution. Hamm forces them to examine their roles in contributing to company problems and to restructure their own work to change the way the company operates. A climate of openness without retribution is fostered, and senior managers listen to messages from all levels of the company, especially messages suggesting improvements in their own work.

> "A sequence of steps, each consisting of decisions made for good reasons, does not necessarily lead to a good result."

Envision a little pig constructing a house made of bricks.

Phase 3: Success

Porcine Products reaps the rewards of this thorough effort. Executives and managers change the way they lead. Cross-company improvements change the way the company operates. Products are created more efficiently and have better quality. Costs go down, orders increase, and morale improves.

Phase 4: Publicity

The business press notices the successes of Porcine Products. Hamm explains the improvements his company has achieved and is asked for a name for his method. In honor of his French grandfather, he calls the improvement Balle-Argentee. The press also wants to report how much time and money was spent, and what was reaped from the improvements; Hamm looks back and makes estimates. From these the business press calculates the magic return-on-investment (ROI) number for the Balle-Argentee method of business improvement.

Envision a little pig proudly holding a book showing a house of bricks on the cover. The book's title is "The Balle-Argentee Method."

Phase 5: Momentum

Other companies look eagerly at the success of Porcine Products. Some of them are experiencing a competitive disadvantage because Porcine Products is now working more effectively than their own company, while others want to achieve the publicized ROI. Discussions at meetings of executives focus on what Porcine Products did, and why it worked.

Phase 6: First Replication

Executives at these other companies decide they want to reproduce Porcine Product's success. They talk with Hamm and others in his company about what actually happened. Each company assigns a senior manager to oversee the implementation of this improvement method across the companies. These senior managers carefully read the literature about the Balle-Argentee method. Implementers look at their own companies' problems and seek to implement the spirit as well as the letter of the Balle-Argentee approach. When they make recommendations, they listen to suggestions for improvement in their own work. They keep close watch on expenditures and benefits of this approach so they will be able to report their ROI.

Envision two or three other little pigs constructing house of wood.

Phase 7: Confirmation

Some of these companies publish studies of their own success using the Balle-

Argentee method. The studies cite the specific improvements each company decided to make. Because of the attention paid to investments and returns since the adoption of Balle-Argentee, this set of companies can cite precise ROI figures. These companies earn accolades from shareholders for fiscally effective management. General business books about this method are published, including "Balle-Argentee in Warp Time," and "Balle-Argentee for Small Companies."

Envision a collection of books with houses of wood on the cover.

Phase 8: Proceduralization

Many more companies decide that this method is valuable. The ROI convinces some, and the fact that their competitors are reaping the returns from Balle-Argentee convinces the rest. With this second set of companies, the executives and senior managers add Balle-Argentee as one more method in their current process initiatives. Because they cannot adequately focus on all of the methods, they delegate implementation of the Balle-Argentee effort to middle managers. These middle managers are given ROI goals that match the published numbers. Other middle managers are given comparable ROI goals for simultaneously implementing different process improvement efforts. Executives believe that the competition engendered by these multiple initiatives will increase the fervor in implementing all the initiatives.

What the implementing managers know about the Balle-Argentee method is limited to the published results. Time constraints prevent these managers from contacting Porcine Products or from reading any of but the shortest summary articles. To reduce the risk of missing their ROI goals, the managers seek ways to improve the cost-effectiveness of Balle-Argentee as they implement it. Implementation that took Porcine Products several years must now be completed within a fiscal cycle. The implementing managers require their people to use some of the specific improvements described in the literature exactly as they are described, without costly discussion or modification. Other specific improvements are ruled out because they would be costly to implement. The stated rationale is that these improvements will not work here because company circumstances differ.

Instead, the implementing managers restate general strategies in the Balle-Argentee literature as broad goals, which they then apply in a sparing manner. In almost all cases, the imperative for executives and managers to listen to workers and to change their own work accordingly is the first general strategy to be deleted. It is restated as *improve communication* and then becomes implemented as *improve communication downward*. These implementing managers have risen in their companies because they respect the wisdom of their superiors. They do not ask for literal implementation of the strategy *executives must listen more* because to do so might cause their superiors to feel threatened or embarrassed.

Finally, these implementing managers seek to cast their own actions in the best light. They believe involving executives would signal weakness. Much of the implementation of Balle-Argentee shifts to managing the news. Executives and senior managers remain uninformed and are uninvolved in the improvement effort except in expecting to reap benefits.

Envision an entire village of houses made of straw.

Phase 9: Diminished Returns

Because of cost cutting, time compression of the improvement effort, lack of executive involvement, dilution of emphasis due to other improvement initiatives, and a tendency to apply the steps as a checklist rather than to seek and fix the company's basic business problems, these more recent Balle-Argentee improvement efforts do not reap the published ROI numbers. This happens broadly across the industry.

Envision the village of straw houses starting to crumble, propped up by sticks and invaded by mice.

Phase 10: Blaming the Method

Workers in these companies feel bombarded by misunderstood management initiatives, and Balle-Argentee is applied intrusively asking for additional work in order to claim compliance. Workers know that the checklists they are being asked to follow and fill out are not solving any real problems. Some attend conferences and complain that the Balle-Argentee method makes companies do stupid things. They cite their experiences, complaining that the Balle-Argentee sponsor does not want to hear about any real problems that are not quickly solved. They complain that checklists and complex documentation substitute for investigation and solutions, and that the intense focus on the ROI severely decreases the investment money for making complex improvements rather than applying Band-Aids.

Coupled with the evidence from Phase 9 that current implementations of Balle-Argentee do not provide good ROI, these very real complaints cause the business press to be ruthless in denigrating Balle-Argentee as a flawed approach. Articles appear advocating slaying the Balle-Argentee monster.

Envision the big bad wolf blowing down the village of straw houses.

Phase 11: Starting Fresh

Mr. Boar, a true improvement-minded executive at Animalia, Inc., decides that no one knows his company like he does. He decides to throw out Balle-Argentee along with all the other silver bullets and takes a close look at Animalia's problems and how to fix them.

Envision a different little pig wiping a bunch of books and drawings off his desk. One of the books has a picture of a house of bricks on the cover.

Morals of the Story

- A sequence of steps, each consisting of decisions made for good reasons, does not necessarily lead to a good result. In the parable, each executive and manager was making good decisions within the constraints established by those higher up. The end result was disastrous.
- For best results, start at Phase 1 and stop at Phase 3.
- Only by really looking at your company's problems can you solve them. Other people's strategies worked for them because the strategies were made for them. If you want to make real improvements, you have to do the work of determining your business problems and applying methods that make sense to fix them.
- Do not assume that people who claim to be using a method really are using it. In Phase 8, actual use of the original method ceases. Instead we have a method bearing the same name that attempts to reap results quickly and dirtily, and is thoroughly unsuccessful in doing so. (Unfortunately, Phase 8 is also where most companies try to use the method.)
- There is nothing like the original. Do not read everyone else's interpretation of a method, read the original. If possible, talk to the creator. Find out the

principles behind the steps, so you can ensure your adaptation is consistent with the principles.
- The ROIs of multiple improvement initiatives do not add; they interfere. Focus on what problems you want to solve, and work out as executives how the initiatives contribute to solutions. Determine where the initiatives will appear as conflicting to the workers and reconcile them. Display a unified front to the workers.
- Do not assume other companies' ROI numbers will apply to you. They started from a different place and made different investments [1].

How to Use Silver Bullets

A great deal has been written about the appropriate way to do process improvement. You must focus on the business goal of improvement, not just on the method used to get there (e.g., CMMI) or on intermediate indicators (e.g., Level 3) [2, 3]. Executives must devote the appropriate resources and stay involved [4]. Managers must learn what is real and react appropriately [5, 6]. The process group must analyze the real causes of problems [7], plan changes, get them approved, and make sure the organization follows through [8]. And everyone must make sure the changes actually improve the product development processes, not interfere with them.

Specific guidance on how to avoid making mistakes with a silver bullet follows:

- **Everyone:** Realize that all methods are a means to the end of an improved company, not the end themselves. You cannot paste on improvements – you have to look at how your company is working right now and how any given method will alter that.
- **Executives:** You have the responsibility to develop your own understanding of what is impeding your company's path to a better future and determining what steps will remove those barriers. Take time to understand any externally generated initiative before pushing it. Does your company even have the same problems as the companies that succeeded with the method?

  Understanding your true problems requires upward communication. Know what currently makes upward communication unsafe, particularly, do your managers have incentives to tell you only good news [9, 10]? Find out what happens to bad news and where it stops. Find a way around that barrier or you will not hear what is really happening in your company.
- **Managers:** You have to understand the method(s) you are pushing. Needing something really badly does not mean you can get it faster by means of wishing or whipping. Determine what is realistic and do not ask for shortcuts.

  Insist on understanding the relationship between initiatives that compete for people's attention. Clarify that relationship, coordinate with other sponsors, and make it easy for the workers to comply with all of the initiatives. Your products make the money, not your management initiatives. You have to be sure the initiatives do not make it difficult to make the product!
- **Process Groups:** Push back on managers demanding the same results as other companies but in less time, with fewer resources, and with less thought. Point out why it is not going to happen. Then propose something that will work. Do not compound the problem by giving only the good news.
- **One Last Word:** A truly successful effort will result when a company develops specific solutions to its specific concerns. Second-generation applications of these methods can work if they are studied to determine where the true benefits came from, and applied intelligently with appropriate investment. But a house of straw is erected when companies believe they can quickly gain return from low-budget adherence to someone else's solutions.◆

Special Credit

Special thanks goes to Cathy Kreyche for her contributions to this article.

References

1. Sheard, Sarah A., and Christopher L. Miller. The Shangri-La of ROI. Proc. of the 10th Annual Symposium of the International Council on Systems Engineering, Minneapolis, MN, July 2000.
2. Sheard, Sarah A. What Is Senior Management Commitment? Proc. of the 11th Annual Symposium of the International Council on Systems Engineering, 2001. Republished in Proc. of the Software Technology Conference, Salt Lake City, UT, 2002.
3. Kaplan, Robert S. "Implementing the Balanced Scorecard at FMC Corporation: An Interview with Larry D. Brady." Harvard Business Review Sept.-Oct. 1993.
4. Gardner, Robert A. "10 Process Improvement Lessons for Leaders." Quality Progress Nov. 2002.
5. Gilb, Tom. "The 10 Most Powerful Principles for Quality in Software and Software Organizations." CrossTalk Nov. 2002.
6. Baxter, Peter. "Focusing Measurements on Managers' Informational Needs." CrossTalk July 2002: 22-25.
7. Card, David. Learning From Our Mistakes With Defect Causal Analysis. Proc. of the International Conference on Software Process Improvement, Adelphi, MD, Nov. 2002.
8. Bowers, Pam. "Raytheon Stands Firm on Benefits of Process Improvement." CrossTalk March 2001: 9-12.
9. Argyris, Chris. Overcoming Organizational Defenses: Facilitating Organizational Learning. Prentice Hall, 1990.
10. Argyris, Chris. Flawed Advice and the Management Trap. New York: Oxford UP, 2000.

About the Author

Sarah A. Sheard is technical lead for Systems Engineering at the Software Productivity Consortium. She has more than 20 years of experience in systems engineering and process improvement. Sheard has published more than 20 articles and papers on systems engineering and process improvement in CrossTalk, the proceedings of software technology conferences, International Council on Systems Engineering (INCOSE) symposiums, and the INCOSE journal. Sheard received INCOSE's "Founder's Award" in 2002. As the consortium's technical lead for the Capability Maturity Model Integration (CMMI®), Sheard was the lead author of the Software Productivity Consortium's course on "Transitioning to the CMMI."

Software Productivity Consortium
Phone: (703) 742-7106
Fax: (703) 742-7350
E-mail: sheard@software.org

Wireless Data Entry Device for Forward Observers

Paul Manz
PM Intelligence and Effects Technology, Inc.

Lt. Col. John R. Surdu, 2nd Lt. Alison M. Adas, 2nd Lt. Zachariah R. Miller,
2nd Lt. Allen J. Peplinski, and 2nd Lt. Erica J. Watson
U. S. Military Academy

The U.S. Army is developing a replacement for the Handheld Terminal Unit to allow forward observers to submit digital calls for indirect fire support. This project explored the use of open-systems platform-independent software on commercial off-the-shelf personal digital assistants (PDA) for this purpose. This article describes the PDA software architecture developed as part of this research.

The purpose of this project was to develop a proof-of-concept Handheld Terminal Unit (HTU) to allow dismounted forward observers (FOs) to create and submit digital calls for indirect fire support (e.g., mortars, artillery, bombs from aircraft, etc.). Feedback from U.S. Army and Marine Corps units, however, indicated that users found the HTU bulky, heavy, and difficult to use. In response, the project manager (PM), Intelligence and Effects contracted the Information Technology and Operations Center (ITOC) and the U. S. Military Academy to conduct a market survey of available commercial off-the-shelf (COTS) solutions to these complaints.

At the time of the survey (1999-2000), no suitable COTS hardware existed; however, the ITOC recommended that the PM develop open-systems platform-independent software so that when personal digital assistant (PDA) technology provided a viable COTS hardware solution, the software would be ready. The purpose of this project, a continuation of previously reported work [1], was to demonstrate the feasibility of such a platform-independent PDA solution.

The proof-of-concept PDA solution developed as part of this research involves a client interface running on a ruggedized PalmOS PDA. The client software is written in a platform-independent, constrained version of Java designed to run on small devices. A Bridge, also written in Java, connects to Army radios via a serial port and to the client via Institute of Electrical and Electronics Engineers (IEEE) 802.11b wireless local area network (LAN) technology.

Due to space constraints, CrossTalk *was not able to publish this article in its entirety. However, it can be viewed in this month's issue on our Web site at <www.stsc.hill.af.mil/crosstalk> along with back issues of* CrossTalk.

Monitoring Progress in Software Development

Joop van der Linden
The Haagse Hogeschool

To manage software development projects, a suitable model of the specific type of project is required. This article presents key performance indicators, a procedure, tables, and some graphs as characteristics of the above-mentioned model. From this, a metric is derived to meet managerial needs for a tool to measure and present project progress. It provides an easy-to-follow picture for both the client and the project manager of how to manage a development project. Percentages of technical completion can be reported periodically in scheduled and actual figures. The use of the metrics is described and illustrated with examples along with a discussion of the possible effects on the behavior of a project-team.

The process of developing software shows some similarities with the process of designing industrial buildings. The approach to an engineering project to build a refinery only differs from a software development project in terms of scale. In both types of projects, using a metric is required to measure progress. It creates a feedback loop that enables pronouncements to be made on the project's development pace. In addition, unexpected changes in the scope of the project can be pointed out at an early stage.

Just like any feedback loop, a metric must also be suitable for reviewing the status of a project at regular intervals. When measuring progress in engineering projects, terms are used such as the percentage of technical completion, the scope of work, and exceeding the completion date. These terms can be used in the same way in a software development project. There are, however, remarkable differences. Usually, in projects for the design, procurement, and construction of a refinery, several hundreds of thousands of man-hours are involved, while the amount of required design drawings, specifications, and details drawings is enormous. Software development projects are considerably smaller.

The deliverables are hardly visible, and a lot of people with a wide range of disciplines are involved. Due to this, the need for a flexible and simple metric is important. Analyzing the work to be done in a software development project gives rise to the construction of a metric. From this it is possible to issue progress and status reports for managing the project without the need to draw up a fully integrated network plan.

Due to space constraints, CrossTalk *was not able to publish this article in its entirety. However, it can be viewed in this month's issue on our Web site at <www.stsc.hill.af.mil/crosstalk> along with back issues of* CrossTalk.

CrossTalk Presents Top 5 Awards at the Software Technology Conference

The winners of CrossTalk's 2002 U.S. Government's Top 5 Quality Software Projects were presented with their awards at the 2003 Software Technology Conference (STC) held recently in Salt Lake City. Individual awards were presented by Joe Jarzombek, deputy director, Software Intensive Systems, Office of the Under Secretary of Defense (Acquisition, Technology and Logistics) – the department sponsoring the contest. H. Bruce Allgood, director, Computer Resources Support Improvement Program at Hill Air Force Base, introduced the representatives from each winning project to more than 2,300 STC conference attendees.

The intent of this search was to recognize outstanding performance of software teams and to promote best practices. These Top 5 project winners were selected from 70 nominations in this second annual event.

CrossTalk begins accepting nominations for the 2003 Top 5 contest after July 1, 2003. Applications will be available at the Web site <www.stsc.hill.af.mil>.◆

The award for the Joint Helmet Mounted Cueing System Software Upgrade (photo left) was accepted by (from left) Phil King, Boeing Integrated Defense Systems, and Lt. Col. A. J. Scott, U.S. Air Force.

Receiving the award for OneSAF Testbed Baseline (photo right) were (from left) Doug Parsons, Beverly Kitaoka, and Bryan Cole of Science Application International Corporation.

The award for Kwajalein Modernization and Remoting (photo left) was received by (from left) Christopher Moulton, Stephen Rejto, and Bill Jaros of the MIT Lincoln Laboratory.

Receiving the award for Defense Civilian Pay System (photo right) were Roger Meece (left) and Sandra Barrineau (center) of the Defense Finance Accounting Service.

The award for the Enhanced Position Location Reporting System was accepted by Hope Miller (left) and Dr. John D. Olsen (center) of Raytheon Company.

Photos by Janna Kay Jensen of CrossTalk.

Representatives from the Top 5 projects took part in a panel discussion during the 2003 Software Technology Conference, including visual presentations and answering questions.

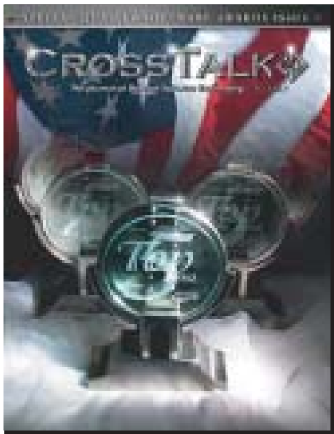
Continued from page 4

Model® Level 3. It looks like the spiral model is also proving to be a success."

Based on these comments, I decided to review the winning nominations again. A few commonalties between the projects surfaced for me as well. The most noticeable was the close conformity to the contracted schedule and budget. This might have been a result of other commonalties such as working hand-in-hand with the users as the software was developed. With most of the projects, the users participated in the requirements definition, answered questions during development, and often were present during all stages of the testing.

Another commonality was a focus on early defect detection. Perhaps the most beneficial commonality between the projects is their uses on multiple platforms, which saves taxpayers millions of dollars.

It seems this year's winners show evidence that the software affecting our lives is being built with a greater degree of success and savings.◆



2003 U.S. Government's Top 5 Quality Software Projects

The Department of Defense and CrossTalk will begin accepting nominations for the 2003 U.S. Government's Top 5 Quality Software Projects after July 1. Outstanding performance of software teams will be recognized and best practices promoted.

These prestigious awards are sponsored by the Office of the Under Secretary of Defense for Acquisition, Technology and Logistics, and are aimed at honoring the best of our government software capabilities and recognizing excellence in software acquisition and development.

The deadline for the 2003 nominations is December 5, 2003. You can review the nomination and selection process, scoring criteria, and nomination criteria by visiting our Web site after July 1. Then, using the nomination form, submit your project for consideration for this prominent award.

ENTER ON OUR WEB SITE AFTER JULY 1

www.stsc.hill.af.mil/crosstalk

CROSSTALK

The Journal of Defense Software Engineering

Get Your Free Subscription

Fill out and send us this form.

OO-ALC / MASE
6022 Fir Ave.
Bldg. 1238
Hill AFB, UT 84056-5820
Fax: (801) 777-8069 DSN: 777-8069
Phone: (801) 775-5555 DSN: 775-5555

Or request online at www.stsc.hill.af.mil

NAME:____________________________

RANK / GRADE:______________________

POSITION / TITLE:____________________

ORGANIZATION:______________________

ADDRESS:__________________________

____

BASE / CITY:________________________

STATE:___________ZIP:______________

PHONE:(____)______________________

FAX:(____)________________________

E-MAIL:____________________________

CHECK BOX(ES) TO REQUEST BACK ISSUES:

MAR2002	□	SOFTWARE BY NUMBERS
MAY2002	□	FORGING THE FUTURE OF DEF.
AUG2002	□	SOFTWARE ACQUISITION
SEP2002	□	TEAM SOFTWARE PROCESS
OCT2002	□	AGILE SOFTWARE DEV.
NOV2002	□	PUBLISHER'S CHOICE
DEC2002	□	YEAR OF ENG. AND SCI.
JAN2003	□	BACK TO BASICS
FEB2003	□	PROGRAMMING LANGUAGES
MAR2003	□	QUALITY IN SOFTWARE
APR2003	□	THE PEOPLE VARIABLE
MAY2003	□	STRATEGIES AND TECH.
JUN2003	□	COMM. & MIL. MEET

To Request Back Issues on Topics Not Listed Above, Please Contact Karen Rasmussen at <karen.rasmussen@hill.af.mil>.

WEB SITES

The Tri-Service Assessment Initiative

http://tai.pica.army.mil

The Tri-Service Assessment Initiative was instituted by the Office of the Under Secretary of Defense (Acquisition, Technology and Logistics) in 1999 to provide Department of Defense (DoD) program managers with an independent, objective review and analysis of software processes, product development and integration. Each assessment is tailored to the program manager's issues, acquisition strategy, and upcoming milestones. The goal is to strengthen DoD software development and shorten acquisition time lines.

American Society for Quality

www.asq.org

The American Society for Quality (ASQ) promotes advanced learning, quality improvement, and a knowledge exchange. ASQ makes its officers and member experts available to inform and advise the U.S. Congress, government agencies, state legislatures, and other groups and individuals on quality-related topics. ASQ also works with the media to provide resources and referrals.

Practical Software and Systems Measurement

www.psmsc.com

Practical Software and Systems Measurement (PSM) is sponsored by the Department of Defense and the U.S. Army. PSM is an information-driven measurement process that addresses the unique technical and business goals of an organization by providing objective information needed to successfully meet cost, schedule, and technical objectives.

Bluetooth

www.bluetooth.com

The Bluetooth Special Interest Group (SIG) is comprised of leaders in the telecommunications, computing, and network industries that are driving the development of a low-cost, short-range wireless specification for connecting mobile products and a qualification program for interoperability. The Bluetooth SIG promoters include 3Com, Ericsson, IBM, Intel, Microsoft, Motorola, Nokia, Toshiba, and hundreds of associate and adopter member companies.

LETTER TO THE EDITOR

Dear **CrossTalk** Editor,

As a software developer, I very much enjoyed Gerald Weinberg's "Destroying Communication and Control in Software Development" (CrossTalk Apr. 2003). Not surprisingly, during the course of a 20-year career I've witnessed – and sometimes been caught up in – the information demolition tactics he described.

A practice I have adopted to counter at least some of these tactics in the testing arena has been to take it on myself to establish a good working relationship between the development and test organizations. By backing the testers to ensure that they have the time and resources necessary to do good testing (which good testers genuinely want to do), they will in turn back the developers to get the time and resources needed to put out a good product (which good developers genuinely want to do).

By eliminating the all-too-typical adversarial relationship between developers and testers, one presents a united front in the pursuit of resources and reasonable expectations that will lead to the production and release of a quality product, and will take away a wedge sometimes used to play one side against the other.

The cynic would suspect that developers' buddying up to testers is a ploy to get them to let things slide now and then. I can't say it doesn't happen, but it's difficult to fake professional respect and mutual support, especially when push comes to shove on a project. The ongoing rewards of a genuinely cooperative effort far outweigh any short-term gains had by wheedling a tester into letting things go, especially when such a charade comes to light and they realize they've been had.

Marc A. Criley, Consultant
Quadrus Corporation

From: David Cook, Consultant, STSC/Shim Enterprise, Inc.
To: STSC Management
Subject: Suggestion for “Bottom 5” Projects

1. As you know, the July CrossTalk issue deals with the 2002 U.S. Government’s Top 5 Quality Software Projects. Also, as you know, selecting the winners was difficult, given the large number of good projects nominated this year. It took several rounds of judging to narrow down the entries. And, once finished, there was not really a ranking from one to five – just a list of Top 5 programs. We should congratulate not only the Top 5 winners, but all the nominated projects – they were all top-notch.

2. To better meet our readers’ needs, I would like to suggest that next year we also have a new competition – the Bottom 5 Software Projects. In an article that I co-wrote with Theron Leishman in the April 2002 CrossTalk (“Requirements Risks Can Drown Software Projects”), we cited a 1995 Department of Defense study that showed that 75 percent of software either didn’t work or was cancelled. While I feel that the actual picture has improved today … well, things are not *that* much better yet.

3. Our readers already know how to build a project that will make the Top 5 – read CrossTalk regularly, talk to our Software Technology Support Center consultants, and learn from the Top 5 projects in this issue. However, we need to come up with criteria for the Bottom 5 next year. I suggest that we give the following criteria to eligible projects and developers.

   First, try every new and improved language, methodology, and tool that every vendor walking in the door has to sell. Listen to all of their claims, and believe them. Don’t talk to previous customers. You can probably shorten your schedule 20 percent by switching life-cycle models to the Extreme Object-Oriented Visual Unified Structured Point Cost Methodology. Then save another 10 percent by switching to the newest version of C Triple Plus Double Sharp. Another 10 percent can come from almost any automated requirements tool (and remember, there is no learning curve to use the new tool!). Now you can optimistically plan on almost 50 percent shorter development time than your last project. Plus, the new tools are guaranteed to decrease testing and integration by almost 50 percent! Since 50 percent + 50 percent = 100 percent, you’re already finished with the project!

   Plus, remember that when using a new tool or methodology, you only need to send one or two of your staff to training; they in turn can train the rest of the team. Send the most experienced geeks you have, regardless of the fact that they have significantly lower people skills than a dead opossum. Also, remember that the technical geeks sent to the training class will remember only the really obscure and complex issues. The basics (such as simply starting the tool) are probably pretty obvious anyway.

   Second, there is no real need to have customer/developer communication. If the developer really needs an interpretation of a requirement, then it should have been asked during the requirements phase. Once requirements have been gathered, work to eliminate those nasty clarification discussions. When the project is delivered, you can then take a lot of time finding out what the customer really wants. Good developers know that verification and validation are much quicker without end-user involvement. In addition, open channels of communication allow customers to develop contingencies if the schedule slips. This makes you look bad.

   Third, make those critical software engineering and computer science decisions without a real software engineer or computer scientist around. Just about any engineer or pilot is truly qualified to make those complex decisions that affect major parts of the schedule and deliverables. In fact, it really helps if those making the critical decisions meet two criteria: they haven’t really used the system for 10 years, and they haven’t really had an up-to-date engineering course since the one they took that covered the basics of vacuum tubes. Don’t think of calling in an outside consultant to help; that might make you look less capable. It’s easier to get the experts to try and fix the problems later than it is to have the experts help you prevent the problems from actually occurring.

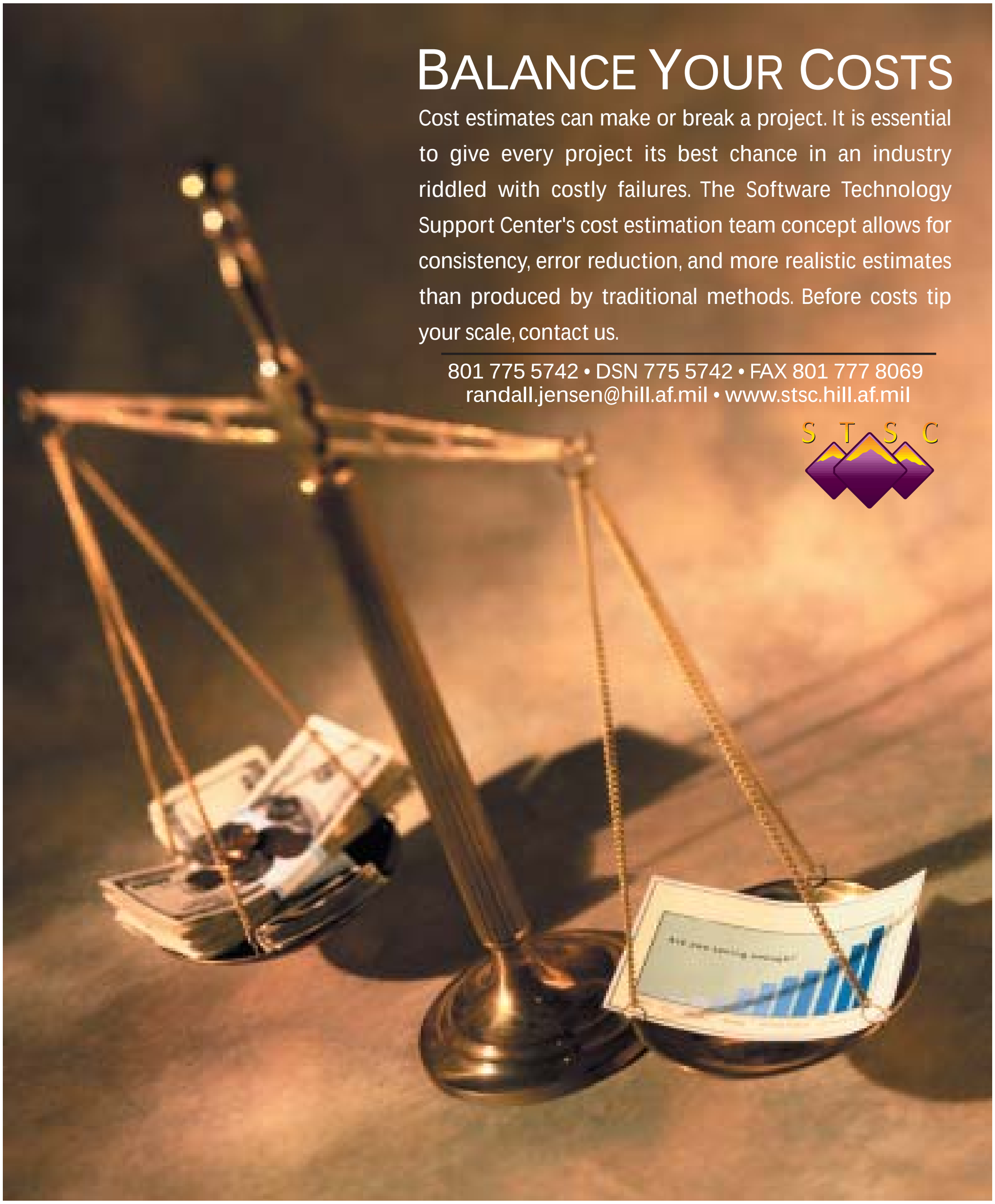
   Fourth, remember that the Capability Maturity Model® (CMM®) and the Capability Maturity Model Integration (CMMI®) are really just paper exercises. We all know that there are really three processes in any serious development effort: the one that is documented and briefed to the CMM/CMMI appraiser, the one that management believes is being used, and the one that you really use. Make sure that the documented process is truly a paperwork exercise. Don’t actually use the documented process; that might slow you down and force you to think about quality way too early. Heck, we all know that quality really starts in the testing phase.

   Finally – this is absolutely critical – trust existing documentation, especially regarding legacy code or systems that are critical interfaces. Trust that the code does what is says it will do, and that the interface works exactly as specified. After all, if you started looking at the interfaces before integration, you might have to change the architecture and design. Just wait until testing, and patch the code as necessary.

4. There are, of course, other criteria for ensuring a projects’ inclusion on the *Bottom 5* next year, but the ones above are usually adequate to ensure that a project qualifies for nomination.

5. As long as projects continue to follow the five criteria above, it will reduce the number of projects that are of sufficient quality to merit nomination for the Top 5 contest. In fact, it almost guarantees that a project can’t be considered for the Top 5 next year. And having fewer projects submitted will make the scoring and selection a lot easier.

– David A. Cook
Software Technology Support Center/Shim Enterprise, Inc.



MISSION CRITICAL SOFTWARE
CRSIP

Sponsored by the
Computer Resources
Support Improvement
Program (CRSIP)

STSC

Published by the
Software Technology
Support Center (STSC)

CrossTalk / MASE
6022 Fir Ave.
Bldg. 1238
Hill AFB, UT 84056-5820