Earned Value Management: Keeping Your Project On Track
How long can you wait for CMMI® Compliance?

Manage your software and systems engineering projects in guaranteed compliance with the CMMI NOW!

processMax 2i and 3i include all the necessary policies, procedures, guidelines, criteria, templates, and forms in role-based, step-by-step instructions, ready for use—everything you need for compliance with the CMMI-DEV at Level 2 and Level 3 respectively. Integrated with robust document management, workflow, and automated measurement and reporting, processMax is the intranet web-based solution for effective and efficient management of your software and systems engineering projects. processMax enables you to start operating in compliance with the CMMI immediately! We guarantee that organizations using processMax will not fail a formal appraisal led by an independent SEI-licensed appraiser.

pragma SYSTEMS CORPORATION
www.pragmasystems.com
877.838.PMAX
info@pragmasystems.com

GSA Schedule Contract No. GS-35F-0559S. processMax is a registered trademark of pragma SYSTEMS CORPORATION. Although processMax makes use of portions of “CMMI” for Development, Version 1.2,” CMU/SEI-2006-TR-008, copyright 2006 by Carnegie Mellon University, neither the Software Engineering Institute nor Carnegie Mellon University have reviewed or endorsed this product.

Copyright © 2008 pragma SYSTEMS CORPORATION.
According to the Department of Defense (DoD) Earned Value Management (EVM) Implementation Guide [EVMIG 2006] EVM is defined as follows: “EVM is a program management tool that integrates the work scope, schedule, and cost parameters of a program, in a manner providing objective performance measurement and management. As work is performed, the corresponding budget value is ‘earned’.” Currently, an EVM System (EVMS) is required for DoD contracts of more than $20 million and, for most DoD contracts of $50 million or more, the company’s EVMS must show validation, which is a formal recognition of certification by an independent party that a company’s EVMS meets certain guidelines. However, EVM can (and in many cases, should) be used on other DoD contracts and software projects in industry. As the articles in this issue will demonstrate, there is a lot of “value” in using EVM on projects.

Basically, EVM uses three parameters to assess the health of a project at any point in time: Planned Value (PV), Actual Cost (AC), and Earned Value (EV). Figure 1 illustrates two of these parameters: PV and AC. PV is the budget assigned to the work scheduled to be accomplished. In EVM, PV is usually estimated in the beginning of the program or project, and is assessed at various points in time, such as at project milestones. For example, in Figure 1, the three month milestone could be a requirements review for a software project, and the six month milestone could be a design review. At the scheduled end of the project (12 months, in this case), the work will be finished, and PV will be the Budget at Completion (BAC), the budget for the entire project ($1 million, in this case). At selected points in time, where PV is identified, the Actual Cost (AC), the amount actually spent, can be compared to PV. At the end of Month 3 in this case, PV is $200K and AC is $230K.

What often occurs now is that program managers and other stakeholders draw conclusions from comparing these two values. It would appear in this case that the project is overrunning by $30K, and we may be in trouble. If the AC was lower than the PV, it could appear that money is not being spent fast enough. In fact, some projects have been cancelled because of “insufficient spend rates” where AC was lower than PV, and a conclusion was (wrongly) drawn that the programs were not managed well. Such isolated comparisons of AC and PV are misleading, however, since we do not know how much work was actually accomplished. The value of the work accomplished to date is the third parameter, Earned Value (EV). Figure 2 shows our project chart with EV added for a third reference point. It shows we not only accomplished more than we planned to, but more than we actually spent to date!

The frequency of assessing project health depends on factors such as the size and complexity of the project, project risks, the development environment and project duration. For smaller projects of short duration, one might choose to monitor EV bi-weekly or monthly. The goal is to select a frequency that will provide visibility into cost and schedule issues while there is still time to resolve potential problems.

Once the three parameters, PV, AC, and EV, have been determined, quite early in the project schedule, some computations can be made to assess the health of the project. Two variances, cost and schedule variance, are first computed as follows:

\[
\text{Cost Variance (CV)} = EV - AC
\]

\[
\text{Schedule Variance (SV)} = EV - PV.
\]

For the example above, CV is +$20K, and SV is +$50K. Positive variances usually mean a project is doing better than expected, and negative variances indicate a worse-than-expected situation. To determine how good or bad the project will be if current trends continue, two other computations can be made:

\[
\text{(Cost) Estimate at Completion} = \text{Budget at Completion} \times \frac{AC}{EV}
\]

\[
\text{(Schedule) Estimate at Completion} = \frac{\text{Planned Schedule}}{\text{PV/EV}}
\]
Here, the Cost Estimate at Completion is $920,000, or an $80,000 under-run. Schedule Estimate at Completion is 9.6 months, or 2.4 months ahead of schedule.

Of course, the assumption is “if current trends continue”; there is no guarantee of continued success. More often, one or both variances are negative, and the project manager may propose a “get well plan” for maintaining control of costs and schedule while accomplishing the work. The real significance of this EVM approach lies in the fact that these estimates can be obtained very early on in the project. It is well established that at 20% into the project these estimates are quite accurate because project productivity seldom increases and typically remains stable over the life of a project.

It should be noted that, as explained by Flemming and Koppelman, EVM provides an early warning signal to project management that a project may be in trouble; however, use of EVM will not in itself prevent problems caused by poor management, inadequate funding, or scope creep [Flemming and Koppelman 2005]. Also, use of EVM does not tell a project manager how to resolve variances; EVM only indicates that the variances exist.

One challenge to software project managers is that a lot of earned value examples are from hardware projects, and involve measures like units produced versus units planned. It is easy to track EV because each unit can be assigned a value and then it becomes a simple task of counting to derive the EV at any point in time. The challenge for software is to break down the scope of work into elements that can then be allocated an appropriate portion of the total budget. The accuracy in doing this determines how successful our EVMS will be in monitoring the state of the project and alerting us to potential cost and schedule problems. One must also define the earning rules up front so that the EV is based on objective criteria, rather than subjective judgment. For example, if a portion of the budget is allocated to requirements analysis, perhaps the criteria for crediting the EV would be based on the number (percentage) of requirements traceable to design, as opposed to a subjective judgment that requirements analysis is, say, 40% complete. There are various schools of thought regarding when EV can be taken. In agile development EV can only be taken when the software feature or functionality is actually implemented in the software and fully tested. The article by John Rusk discusses the rationale for this rule. Since the 1990s, some viable measures of EV have been suggested for software. Some of these measures were suggested from an Air Force Institute of Technology thesis effort [Ayres and Rock 1992], and later published as part of a journal article [Christensen and Ferens 1995]. They include:

1. Requirements and Design Progress: The number of planned versus actual requirements completed and, later, the number of planned versus actual component designs completed. This could be useful in assessing PV and EV.

2. Code and Test Progress: The number of planned versus actual units coded and, later, tested. This also could be useful in assessing PV and EV, albeit later in the program.

3. Person-months of Effort Expended. This is useful in assessing AC.

Let’s say we use a cost estimating model to estimate the cost of a software project. (This could be a single project, the first iteration or spiral of an evolutionary development project, or a major task within a larger project), and the estimates from a cost model are as follows:

| Requirements Analysis: 3 Months, | $200K |
| Design: 3 Months, | $300K |
| Code and Test: 6 Months, | $500K |
| Total: 12 Months, | $1M |

These estimates are plotted in Figure 1 as PV at each schedule milestone. During planning for this project, we also determine we should have 40 requirements defined by the end of Month 3, and 60 components designed by the end of Month 6. For this example let’s assume that all requirements and components are roughly equal and we will assign a value of $5K for each requirement defined and, later, $5K for each component designed. In a real world project this process of assigning value is much more complex. Our plan indicates that at the end of Month 3, we can expect to spend $200K and have all 40 requirements defined. Suppose we then use the “Person-months of Effort Expended” measure to assess AC which, in this example, is $230K.

Let’s say that, at the end of Month 3, we have all our requirements defined ($5k x 40 = $200k), and we have designed 10 components ($5k x 10 = $50k). As shown in Figure 2, the total EV for the project at that time is $250K. We have done $50K more work than we had planned, even though we have spent $30K more than planned. The results are positive variances in both cost and schedule. If we want to find out why we are doing so well, there are various measures we can consider. In reality, the project manager would investigate the reasons...
behind the variances and be able to explain the causes; however, since positive variances indicate a healthy project, the manager would probably not spend a lot of time investigating unless one or more of the variances drops to a negative value.

Today, there are many resources available for software measures that can be useful in EVM, either as direct measures of PV, AC, and EV, or indicators that can be used to help explain cost and schedule variances. The Practical Software and Systems Measurement (PSM) Guide [PSM 2003] includes more than 50 measures that may be useful (including EV itself), and Capers Jones [Jones 2008] describes a multitude of potentially useful measures. Practitioners [Christensen and Ferens 1995] also describe key measures for explaining variances.

This is not to say that Earned Value for software is an easy undertaking. An organization must be committed to tracking EV measurements, as well as AC. Also, PV for software requires accurate software size, effort, and schedule estimating, which is not always easy to do (See STN Volume 11, Issue 3). A good work breakdown structure (or some other work decomposition method) is needed to determine what each separate software project task entails and how much for the overall time and effort will be needed for it.

The articles in this issue will help motivate readers to use EVM, explain how EVM can be used for software projects, and provide methods to use EVM. The first article, by Wayne Abba, one of the world’s best known experts in EVM, examines Government Policy for EVM, from its inception in the 1960s to the present. He comments on how government policy shifts affect the value of EVM as a management tool. The next two articles address use of EVM for software projects. The article by Hunt, Solomon, and Galorath shows how to specifically address EVM for software projects, including using Solomon’s widely-published Performance-Based Earned Value methods. The article by John Rusk, a software architect from New Zealand, shows how EVM can be applied to agile software development efforts. In the next article, David Bachman, the EVM curriculum director for Defense Acquisition University (DAU), explains how EVM artifacts and estimates are used in DAU’s 12-Step Integrated Program Management Model for decision making, action planning and addressing program risks. This article includes a reference to an example for using the model on an actual program.

In addition to these EVM theme based articles, DACS presents an article (by multiple authors) that describes the SPRUCE initiative. SPRUCE, which stands for Systems and Software Productivity Collaboration and Experimentation Environment, is funded by the Office of Secretary of Defense (OSD) and is technically guided by the Air Force Research Laboratory (AFRL) at Rome, NY. It is an open web portal to bring together DoD software developers, users, and software engineering researchers virtually by enabling their collaboration on specifying and solving software producibility challenge problems from real-world scenarios.

References


About the Authors

Dan Ferens works for the ITT DACS as an analyst and as an instructor for a 12-part series in software affordability which has been taught mainly to Air Force Research Laboratory (AFRL) scientists, engineers, and managers in Rome, NY. Dan retired from AFRL in early 2007 after more than 35 years of service to the Air Force as a military and civilian employee. Dan has been involved in software estimating since he became a civilian in 1978, both as an AFRL analyst and program manager, and as a Professor at Air Force Institute of Technology where he taught classes on software estimation and other software engineering and management topics for 13 years. He is currently an Adjunct Instructor at SUNY Institute of Technology in Utica, New York where he teaches a class in information technology project management. He is a life member of the International Society of Parametric Analysts (ISPA), where, in 1999, he received the prestigious Freiman award for lifetime achievements in parametric estimating. He is also a member of Toastmasters, International where he holds the rank of Distinguished Toastmaster. Mr. Ferens has a Masters degree in Electrical Engineering from Rensselaer Polytechnic Institute, and a Masters Degree in Business Administration from the University of Northern Colorado. He and his wife, Marcie, currently reside in Fulton, New York.

Ellen Walker, is a DACS Analyst, currently developing a series of publications on software “best practices” as part of the DACS Gold Practice Initiative. She has spent the past 25 years as a software developer in various roles spanning the entire software life cycle including project management of multiple business process re-engineering efforts within the DoD community. She is also experienced with assessment initiatives such as the Capability Maturity Model for Software (CMM-SW) and the quality management practices of the New York State Quality Award program. Ellen has an MS in Management Science (State University of New York (SUNY) at Binghamton), and bachelor degrees in both Computer Science (SUNY – Utica/Rome) and Mathematics (LeMoyne College).

Author Contact Information
Email: Daniel Ferens [daniel.ferens@itt.com]
Email: Ellen Walker [ellen.walker@itt.com]
Software quality and reliability are mission critical. The size, pervasiveness, and complexity of today's software can push your delivery dates and budgets to the edge. That means rigorous quality discipline, and collaboration between development and QA teams has never been more important.

With offices worldwide and over 8,500 customers, Seapine Software is the leading provider of integrated quality-centric application lifecycle management (ALM) solutions. For over 13 years Seapine's easy-to-use, award-winning tools have helped companies achieve success by streamlining communications, improving traceability, and making development and QA teams more productive.

Satisfy Your Quality Obsession.
www.seapine.com/gsa

GSA Advantage® GSA Schedule 70 Contract GS-35F-0168U
Integrated ALM for a Quality-Ready Advantage

©2008 Seapine Software, Inc., Seapine, and the Seapine logo are trademarks of Seapine Software, Inc. All Rights Reserved.
Government program management policies and processes are subject to change, especially when a new presidential administration’s management philosophy differs sharply from its predecessor. Public servants are responsible for implementing the changes as effectively and efficiently as possible. But as the changes filter through the many levels of political and career managers in the bureaucracy, “effectiveness and efficiency” are subject to multiple interpretations.

As the senior program analyst for contract performance management in the Office of the Secretary of Defense from 1982-99, I served under three presidents, six secretaries of defense, and twelve principal executives (all political appointees – comptrollers and under secretaries of defense). Political appointees’ tenures on average are brief and development cycles for defense programs are long, just one of many factors that contribute to seesaw policy swings. Effective executives achieve good results by fostering dialogue with their career staff and stakeholders both within and outside government to understand what is working in their area of responsibility and what needs improvement.

Throughout my years in the Pentagon I was the senior OSD analyst for contract performance measurement, reporting to Mr. Gary Christle, a career member of the Senior Executive Service who enjoyed excellent relationships with his career and political superiors and received several top-rank SES awards. Mr. Christle guided the evolution of Earned Value Management through numerous changes, both externally and internally driven.

When established as DoD policy in 1967, EVM was meant to be the integrating tool for technical, cost and schedule performance measurement of major defense contracts. But EVM implementation exceeded its intended purpose as time passed. Policy became disconnected from practice. Earned value came to be regarded both in government and industry as a contract financial reporting requirement, with increasingly stringent processes demanded by DoD review teams that audited the contractors. In the early 1990s Mr. Christle initiated a series of reforms designed to restore management value to EVM.

The implementing guidance for the “three-legged stool” supporting DoD program management requirements included MIL-STD 499 for systems engineering (the technical leg), and DoD instructions governing cost estimating and EVM and its associated performance reports (the cost leg) and contract funds reporting (the financial leg). The cost and financial pieces, along with quarterly reporting to the Congress in the Selected Acquisition Report, were known collectively as “Selected Acquisition Information and Management Systems.”

Policy and oversight for the functions represented by the three legs belonged to three different organizations. Systems engineering was in acquisition, cost estimating was in Program Analysis and Evaluation’s Cost Analysis Improvement Group, and cost and financial management and reporting requirements were assigned to Mr. Christle’s office in the Defense Comptroller (later transferred to the acquisition office).

This article discusses the major changes to EVM from its DoD origins to the beginning days of the Obama Administration, with EVM now required for all agencies. More than forty years after becoming Defense policy, EVM experience is mixed. The DoD management stool is not just wobbly, it’s broken. The technical leg is missing, the financial leg is weak and the cost leg is being reconfigured as an audit leg – a trend that, if not reined in, threatens to further erode EVM’s usefulness as a proven management concept. Meanwhile, the civilian agencies are improving their management processes in response to Office of Management and Budget requirements and challenging DoD’s reputation as the leading EVM authority.

What Happened to Technical Performance?
The Clinton-Gore “Reinventing Government” agenda mandated new ways of doing business in all government agencies. Its aptly named “Hammer” award rewarded regulatory
destruction on a massive scale. The Pentagon created an Acquisition Reform organization and cancelled thousands of military specifications and standards. Among them were two that governed disciplines essential to EVM-based management – MIL-STD-499 for systems engineering and MIL-STD-881 for work breakdown structures. My office owned WBS policy.

With the regulatory environment laid waste by the mass extinction of specs and standards, policy owners faced stark choices. We could appeal to the Defense Standards Improvement Council to retain the documents, convert them to another form or let them die. Because a standard program WBS is essential to define the program and to provide a framework for systematic cost collection and for setting up an EVM system, my office pursued the conversion alternative. In those heady reform days, retention simply was not an option. We didn’t waste time – ours was the first case heard by the council.

The standards council approved our proposal to create a military handbook, MIL-HDBK-881, that preserved WBS as a policy requirement at the program level and that provided guidance for contractors to extend the contract WBS. In the recurrent struggle to strike a balance between specific requirements vs. general guidance, the OSD Acquisition Reform office emphasized high-level guidance.

The systems engineering proponents took a different tack, arguing that MIL-STD-499 should be retained. When asked by the Deputy Under Secretary of Defense for Acquisition Reform why the Department needed the standard, they replied that defense contractors do not do systems engineering well. She countered with the pointed question: “Why in the world would you want to contract with such a company?”

Why, indeed? A decade later, with perfect hindsight, the situation is clearer. Failure to sustain a robust DoD systems engineering standard and the lack of effective competition as the defense industrial base shrank combined in a perfect storm. Auditors cite inadequate systems engineering as a key contributor to program failures. In the vacuum created by loss of the standard, defense EVM policy alone, with brief allusions to technical performance integration, was insufficient to maintain the balance. To restore it, the recently approved DoD Instruction 5000.02, Operation of the Defense Acquisition System, includes greatly strengthened systems engineering requirements.

EVM Reform

Defense Acquisition Reform initiatives helped to improve EVM policy and implementation. Mr. Christle did not dispute an Acquisition Reform-sponsored study finding that EVM contributed 0.9% to the 18% non-value-added cost premium that the study identified for companies doing business with DoD. However, he was able to show that fully two thirds of the 0.9% was caused by excessive EVM implementation requirements. Good management is not free. Of course it should not be excessive, but any “cost of management” pales next to the cost of program failure.

For example, the failure of the Navy’s A-12 Avenger II aircraft development program also fueled the reform initiative. Failure and crisis spark behavioral changes in government and the A-12 provided both to unprecedented degrees when the multi-billion dollar contract was terminated for default in 1991. Mired in litigation and appeals since then, the A-12 is the largest contract termination case in history – settlement costs to the eventual loser are estimated at $2.7 Billion and growing – and a profound example of EVM information not being used by industry and government managers.

By restoring EVM’s value to management and working with all stakeholders – the military departments, industry and the EVM community – measurable improvements were made to defense management during the 1990s. The A-12 successor, the F/A-18E/F “Super Hornet,” pioneered government-industry teamwork, EVM and other management reforms that contributed to the program’s exceptional acquisition success and helped bring the Department’s EVM reforms to the attention of OMB.

Leaders at OMB were surveying government agencies for best management practices to implement statutory performance requirements including the Government Performance and Results Act and the Federal Acquisition Streamlining Act. They decided that DoD was in the lead and incorporated defense EVM principles in OMB Circular A-11, Part 3 (now Part 7),

Its promise is achieved when it is viewed primarily as a management tool. When that ideal is lost or marginalized, EVM tends to devolve to an excessive reporting and oversight requirement that further diminishes its value for management – a vicious cycle that harms management in the long run.

By the end of the millennium, DoD had built a new foundation for EVM-based program management. Unfortunately it did not prove to be durable. After Mr. Christie retired from public service in 2000, his successors did not build on that foundation and reduced the level of collaboration with industry. The emphasis on management was replaced by reliance on data reporting – a return to the discredited notion that EVM is about contract reporting and oversight, not management.

Given the lack of management attention, EVM regressed. Before long, defense contracts again suffered “surprise” cost and schedule problems that attracted scathing attention from DoD and Government Accountability Office auditors. In response, DoD reinvigorated its contractor review process by beefing up the audit staff at the Defense Contract Management Agency. The auditors are resurrecting the traits of the pre-acquisition reform era: unilaterally interpreted, excessively detailed, non-value-added requirements – an overcorrection that is harming project management and government-industry relations.

A memorandum signed by the Under Secretary of Defense (Acquisition, Technology & Logistics) on January 24, 2009 may herald a long overdue correction. It assigns primary EVM policy to the Deputy Under Secretary of Defense (Acquisition & Technology) and establishes a Defense Support Team chaired by the Director of Defense Procurement and Acquisition Policy.

The team will include senior executives and flag officer representatives from OSD, DCMA, Defense Acquisition University and the military departments. Along with appropriate input from defense industry, formation of this team can go a long way toward ensuring that all stakeholders’ views are considered in formulating the policy and guidance changes that are needed to restore balance to EVM implementation.

A Word on EVM and Software

Guidance from OMB requiring application of EVM-based management to capital investment programs includes software development programs. The emphasis stems from the Information Technology Management Reform Act (Clinger-Cohen) and from strong leadership by directors of OMB’s Office of E-Government & Information Technology. No longer considered to be so “different” that normal project management disciplines do not apply, IT management will continue to benefit from incorporation into the project management mainstream.

The Growing Role of GAO

On March 2, 2009, the Government Accountability Office released the “GAO Cost Estimating and Assessment Guide: Best Practices for Developing and Managing Capital Program Costs.” The guide was developed initially as a guide for auditors but has evolved into a resource for agencies that lack the cost estimating staff of large agencies like DoD and NASA.

The GAO guide is a comprehensive description of best practices for integrating the cost estimating, risk management and EVM disciplines. OMB Circular A-11 Part 7 and the Capital Programming Guide point agencies to the GAO guide, making this excellent new document a vital resource for the entire government – Executive Branch agencies and auditors alike. It is available at www.gao.gov – enter “GAO-09-3SP” into the search block.

Conclusion

“Change is not made without inconvenience, even from worse to better.” — Samuel Johnson 1709-1784, English lexicographer

Through more than forty years, earned value management has proved itself as the best way known to measure and manage integrated technical, cost and schedule performance – first on defense contracts and more recently on all government programs. Its promise is achieved when it is viewed primarily as a management tool. When that ideal is lost or marginalized, EVM tends to devolve to an excessive reporting and oversight requirement that further diminishes its value for management – a vicious cycle that harms management in the long run.

The state of EVM in DoD has been deteriorating for several years. Some would argue that the heightened oversight of contractors by DCMA will improve the situation. I disagree. No doubt the “accuracy” of contractor reporting will improve, but at the expense of management – in other words, DoD is repeating the mistakes that led to the wave of reforms undertaken during the Clinton-Gore era, and ignoring the lessons learned from Acquisition Reform.

Can the balance be restored? I believe the answer is a qualified “yes.” If the new DoD EVM policy is informed by an open dialogue – if DoD leaders are willing to reverse the excesses being perpetrated in the name of EVM – if OMB and GAO efforts are synergistic – we can indeed right the balance. All the pieces are in place to support the Obama Administration’s call for more openness, transparency and accountability in government. This time, we can get it right.
About the Author

Wayne Abba is an Earned Value Management expert who teams with government entities and the private sector. He also is a part-time member of the Research Staff at the Center for Naval Analyses.

Mr. Abba retired from the Office of the Secretary of Defense in 1999. He is a principal author of National Defense Industrial Association management guides that were adopted by the US Office of Management and Budget for use by government agencies and their contractors. As president of PMI’s College of Performance Management in 2002-2003, Mr. Abba began development of the Practice Standard for Earned Value Management.


Author Contact Information
Email: Wayne Abba [abbaconsulting@cox.net]

The DACS website has been updated with new research information.

The DACS is a central distribution hub for software technology information sources. The DACS offers a wide-variety of Technical Services designed to support the development, testing, validation, and transitioning of Software Engineering technology.

Top : SEI Capability Maturity Model Integration (CMMI)

This Software Engineering Institute’s project’s objective is to develop a product suite that provides industry and government with a set of integrated products to support process and product improvement.

- Education and Training (17) - Education and Training Courses for the CMMI
- Literature (2) - Books, papers, reports, and articles that contain useful information about the CMMI and its predecessor, the CSCI
- Programs and Organizations (7) - Groups, programs and organizations that focus on the CMMI
- Service Providers/Consultants (1) - Service providers and consultants for the CMMI model
- Standards, Policies, and Procedures (4) - Government and Commercial Standards, Policies, and Procedures that apply to the CMMI
- Tools (3) - Software tools, spreadsheets and utilities that support SEI Capability Maturity Model Integration (CMMI)

Be sure to check it out at https://www.thedacs.com/
Applying Earned Value Management to Software Intensive Programs

INACCURATE ESTIMATES CAN THREATEN PROJECT SUCCESS CAUSING POOR PROJECT IMPLEMENTATIONS, THE SHORTCUTTING OF CRITICAL PROCESSES, AND EMERGENCY STAFFING TO RECOVER SCHEDULE

by Robert P. Hunt and Dan Galorath (Galorath Incorporated), and Paul J. Solomon (Performance-Based Earned Value)

Often, traditional earned value approaches do not deal sufficiently with the idiosyncrasies of software intensive programs. However, successful management of software intensive programs can be achieved by focusing on establishing the requirements, developing a reliable baseline estimate for cost and schedule, selecting effective software metrics, applying Performance-Based Earned Value® (PBEV), and using analytic processes to project cost and schedule based on actual performance.

The Department of Defense estimates that software now accounts for 40% of all research, development, test and evaluation (RDT&E) spending. Software intensive projects that achieve their original cost and schedule projections are rare. Many information technology projects have been declared a disaster area by commercial and government managers. These projects have been too costly, too late, and often don’t work right. Applying appropriate technical and management techniques can significantly improve the current situation.

Inaccurate estimates can threaten project success causing poor project implementations, the short-cutting of critical processes, and emergency staffing to recover schedule. The lack of well defined project requirements and specifications may result in significant growth in cost and schedule. Symptoms of this growth may include constantly changing project goals, frustration, customer dissatisfaction, cost overruns, missed schedules, and the failure of a project to meet its objectives.

The Project Management Institute (PMI) published an analysis of several government defense and intelligence agency large-scale acquisition programs that experienced significant cost and schedule growth. This analysis shows that several critical factors need to be addressed in the pre-acquisition phase of the acquisition cycle. The principal causes of growth on these large-scale programs can be traced to several causes related to overzealous advocacy, immature technology, lack of corporate technology roadmaps, requirements instability, ineffective acquisition strategy, unrealistic program baselines, inadequate systems engineering, and work-force issues.

This paper will discuss some key elements associated with:

1. Establishing a process for requirements definition and developing the cost and schedule baseline
2. Developing a reliable cost and schedule baseline,
3. Identifying critical software management metrics,
4. Applying Performance-Based Earned Value (PBEV), and
5. Using an analytic process (such as SEER Control; formerly called Parametric Project Monitoring and Control (PPMC)) to project cost and schedule based on actual performance.

Establishing a Process for Requirements Definition and Developing the Technical, Cost and Schedule Baselines

A software program life cycle cost estimate is the most knowledgeable statement one can make at a particular point in time regarding effort/cost, schedule, staffing, risk, and reliability. However, the most important business decisions about a software project are often made at the time of minimum knowledge and maximum uncertainty. Cost estimators recognize that the estimate is not a point, but rather a well formed estimate defined

1 Page 134, Trillions For Military Technology; John A. Alic, Palgrave MacMillian, 2007
by a probability distribution.

A well defined process is critical to defining the requirements and completing the initial cost and schedule estimate. The proper use of PBEV provides for integration of project technical scope, schedule, and cost objectives; and the establishment of a baseline plan for performance measurement. Additionally, the use of an analytic tool to project likely cost and schedule based on actual performance provides for realistic projections of future performance. Success of the project can be aided by defining the best objectives, by planning resources and costs which are directly related to those objectives, by measuring accomplishments objectively against the plan, by identifying performance trends and problems as early as possible, and by taking timely corrective actions.

A CMMI tutorial recognizes that people, process, and technology are major determinants of product cost, schedule, and quality. We all realize the importance of having a motivated, quality work force but even our finest people can't perform at their best when the process is not understood or not operating at its best. Figure 1, “People, Process, Technology are Keys”, presents this concept.

In the book, “Software Sizing, Estimation and Risk Management” (Dan Galorath and Michael Evans, 2007) a ten step process is presented for program requirements generations and estimation. Figure 2, 10 Step Software Process, outlines the ten steps. While the Galorath process includes ten steps, other processes may include more or less steps (e.g. the GAO Cost Guide includes a 12 steps process). Note specifically the importance of step 4, estimating and validating the software size metric. The key here is to establish an auditable, repeatable set of steps to establish the requirements and develop the baseline estimate of cost and schedule. This is the key to articulating an accurate requirement and establishing a reliable baseline for cost and schedule.

**Identifying critical software management metrics**

That most large software programs get into trouble is a demonstrated phenomenon. Therefore selecting the correct set of software metrics to track is critical to program success. Practical Software Measurement (McGarry, Card, Jones; Addison-Wesley, 2002) identifies seven information categories and then expands these information categories into measurable concepts and then prospective metrics. This taxonomy is presented in the Figure 3 on next page, “What To Measure.”

Collecting and tracking data on all the prospective metrics is impractical for typical software intensive programs. Software developers often produce their software deliverables in unique environments and with unique processes. In selecting the
appropriate software metrics, the analyst must “do your own thing, but carefully.”

For Earned Value purposes, the most effective software metrics are those that relate to product size, schedule, quality, and progress. For software intensive programs, measures of quantity (e.g., number of lines of code completed) do not accurately reflect the quality aspects of the work performed on neither the program nor the actual progress since items such as lines of code completed do not capture items such as integration, testing, etc.

Size is often measured as Source Lines of Code (SLOC) or Function Points and used as a sizing measure for budgets and for earned value using a percent of completion method. There are two critical problems with this approach. First, there has traditionally been a significant error in estimating SLOC. And, the number of lines of code completed does not necessarily reflect the quality or total progress toward a performance goal. Therefore, any progress metric based solely on SLOC is highly volatile. Whether SLOC, function points, Use Cases, or some other size artifact is selected, a careful process must be utilized to establish a credible size metric. It is recommended that in addition to tracking progress toward a goal, size growth should also be tracked.

Schedule metrics and procedures normally relate to completion milestones, which are also a common tracking metric. Sometimes these milestone definitions and completion criteria lack quantifiable objectives. Often an incremental build is released that does not incorporate all the planned functional requirements or a developer claims victory after just testing the nominal cases.

Progress metrics can be very difficult for large software programs. It is generally agreed that no software is delivered defect free. Software engineers have hoped that new languages and new processes would greatly reduce the number of delivered defects. However, this has not been the case. Software is still delivered with a significant number of defects. Capers Jones estimates that there are about 5 bugs per Function Point created during Development\(^5\). Watts Humphrey found “… even experienced software engineers normally inject 100 or more defects per KSLOC\(^6\). Capers Jones says, “A series of studies show the defect density of software ranges from 49.5 to 94.5 errors per thousand lines of code\(^7\).” The physical and practical limitations of software testing (the only way to determine if a program will work is to write the code and run it) ensure that large programs will be released with undetected errors. Therefore, defects discovery and removal is a key metric for

---

\(^4\) P. 68, A Practical Guide to Earned Value Project Management, Charles Budd and Charlene Budd; Management Concepts, 2005

---

**Figure 3: What To Measure**

<table>
<thead>
<tr>
<th>Information Category</th>
<th>Measurable Concepts</th>
<th>Information Category Measure Mapping*</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Schedule and Progress</td>
<td>Milestone completion</td>
<td>Milestone Dates</td>
</tr>
<tr>
<td>Critical Path Performance</td>
<td>Requirements Traced, Requirements Tested, Problem Reports Opened, Problem Reports Closed, Reviews Completed, Change Requests Opened, Change Requests Resolved, Units Designed, Units Coded, Units Integrated, Test Cases Attempted, Test Cases Passed, Action Items Opened, Action Items Completed</td>
<td></td>
</tr>
<tr>
<td>Work Unit Progress</td>
<td>Components Integrated, Functionality Integrated</td>
<td></td>
</tr>
</tbody>
</table>
| Incremental Capacity |...

---

<table>
<thead>
<tr>
<th>Information Category</th>
<th>Measurable Concepts</th>
<th>Information Category Measure Mapping*</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 Resources and Cost</td>
<td>Personnel Effort</td>
<td>Staff Level, Development Effort, Experience Level, Staff Turnover</td>
</tr>
<tr>
<td>Financial</td>
<td>BCWS, BCWP, ACWP, Budget, Cost</td>
<td></td>
</tr>
<tr>
<td>Environmental/Support</td>
<td>Quality Needed, Quality Available, Time Available, Time Used</td>
<td></td>
</tr>
<tr>
<td>3 Product Size &amp; Stability</td>
<td>Physical Size/Stability</td>
<td>Database Size, Components, Interfaces, LOC</td>
</tr>
<tr>
<td>Functional Size</td>
<td>Requirements, Function Changes, Function Points</td>
<td></td>
</tr>
<tr>
<td>4 Product Quality</td>
<td>Functional Correctness</td>
<td>Defects, Age of Defects, Technical Performance</td>
</tr>
<tr>
<td>Maintainability</td>
<td>Time to Release, Cyclomatic Complexity</td>
<td></td>
</tr>
<tr>
<td>Efficiency</td>
<td>Utilization, Throughput, Response Time</td>
<td></td>
</tr>
<tr>
<td>Portability</td>
<td>Stand Comp-Dilance</td>
<td></td>
</tr>
<tr>
<td>Usability</td>
<td>Operator Errors</td>
<td></td>
</tr>
<tr>
<td>Reliability</td>
<td>MTTF</td>
<td></td>
</tr>
<tr>
<td>5 Process Performance</td>
<td>Process Compliance</td>
<td>Reference Maturity Rating, Process Audit Findings</td>
</tr>
<tr>
<td>Process Efficiency</td>
<td>Productivity, Cycle Time</td>
<td></td>
</tr>
<tr>
<td>Process Effectiveness</td>
<td>Defects Contained, Defects Escaping, Rework Effort, Rework Components</td>
<td></td>
</tr>
<tr>
<td>6 Technology Effectiveness</td>
<td>Technology Suitability</td>
<td>Requirements Coverage</td>
</tr>
<tr>
<td>Technology Volatility</td>
<td>Baseline Changes</td>
<td></td>
</tr>
<tr>
<td>7 Customer Satisfaction</td>
<td>Customer Feedback</td>
<td>Satisfaction Rating, Award Fee</td>
</tr>
<tr>
<td>Customer Support</td>
<td>Request for Support, Support Time</td>
<td></td>
</tr>
</tbody>
</table>

* Practical Software Measurement; McGarry, Card, Jones; Addison-Wesley, 2002
assessing program quality.

The analyst should review the list of potential measures defined in Figure 3, What To Measure, and select the set of metrics that are most appropriate for a specific program.

Applying Performance-Based Earned Value (PBEV)

Performance-Based Earned Value® (PBEV) is an enhancement to the Earned Value Management Systems (EVMS) standard. PBEV overcomes the standard’s shortcomings with regard to measuring technical performance and quality (quality gap). PBEV is based on standards and models for systems engineering, software engineering, and project management that emphasize quality. The distinguishing feature of PBEV is its focus on the customer requirements. PBEV provides principles and guidance for cost effective processes that specify the most effective measures of cost, schedule, and product quality performance.

Program managers expect accurate reporting of integrated cost, schedule, and technical performance when the supplier’s EVMS procedure complies with the EVMS Standard. However, EVM data will be reliable and accurate only if the following occurs:

- The indicated quality of the evolving product is measured.
- The right base measures of technical performance are selected.
- Progress is objectively assessed.

Using EVM also incurs significant costs. However, if you are measuring the wrong things or not measuring the right way, then EVM may be more costly to administer and may provide less management value.

Because of the quality gap in the EVMS standard, there is no assurance that the reported earned value (EV) is based on product metrics and on the evolving product quality. First, the EVMS standard states that EV is a measurement of the quantity of work accomplished and that the quality and technical content of work performed are controlled by other processes. A software manager should ensure that EV is also a measurement of the product quality and technical maturity of the evolving work products instead of just the quantity of work accomplished. Second, the EVMS principles address only the project work scope. EVMS ignores the product scope and product requirements. Third, the EVMS standard does not require precise, quantifiable measures of progress. It states that objective EV methods are preferred but it also states that management assessment (subjective) may be used. In contrast, other standards specify objective measurement. Fourth, EVM is perceived to be a risk management tool. However, EVMS was not designed to manage risk and provides no guidance on the subject.

PBEV is a set of principles and guidelines that specify the most effective measures of cost, schedule, and product quality performance. It has several characteristics that distinguish it from traditional EVMS:

- Plan is driven by product quality requirements.
- Focuses on technical maturity and quality, in addition to work.
- Focuses on progress toward meeting success criteria of technical reviews.
- Adheres to standards and models for systems engineering, software engineering, and project management.
- Provides smart work package planning.
- Enables insightful variance analysis.
- Ensures a lean and cost-effective approach.
- Enables scalable scope and complexity depending on risk.
- Integrates risk management activities with the performance measurement baseline.
- Integrates risk management outcomes with the Estimate at Completion.

PBEV augments EVMS with four additional principles and 16 additional guidelines.

The following are PBEV principles that set it apart from EVMS:

1. Product Scope and Quality. Integrate product scope and quality requirements into the performance measurement baseline.
2. Measure Quality. Specify performance toward satisfying product quality requirements as a base measure of earned value.

3. Integrate Risk. Integrate risk management with EVM.

4. Tailored Application. Tailor the application of PBEV according to the risk.

Figure 4, EVMS and PBEV Flow Chart, overlays the “Performance-Based” process flow on the traditional “Earned Value” process flow.\(^\text{10}\)

PBEV supplements traditional EVMS with the best practices. Its principles and guidelines enable true integration of project cost, schedule, and technical performance. The distinguishing feature of PBEV is its focus on the customer requirements. Measures of product scope and product quality are incorporated into the project plan. Progress is measured against a plan to fulfill all customer requirements. Measuring the wrong things does not dilute management attention. Consequently, management is able to take rapid corrective actions on deviations that threaten customer satisfaction and business enterprise objectives.

Using an analytic process to project cost and schedule based on actual performance

Once the requirement definition is complete; the cost and schedule baseline has been established; the appropriate metrics have been selected; and a PBEV system is in place, the final challenge is to implement a process that quickly and accurately estimates final cost and schedule based on actual performance. This analysis is best accomplished using an analytic/parametric process. Galorath Incorporated calls this process SEER Control (formerly Parametric Project Management and Control. The purpose of SEER Control is to provide an understanding of the project’s progress so that appropriate corrective actions can be taken when the project’s performance deviates significantly from the plan. SEER Control applies a four-dimensional (4-D) approach for assigning progress to the development of each program/application that is part of the project. The first dimension is Software Development Life Cycle (SDLC) primary activity completion for the development of a specific program/application. Each SDLC primary activity, in turn, is assigned progress according to a weighted combination of three other dimensions: artifact completion, milestone completion, and defect discovery/removal. SEER Control provides an “at-a-glance” indication of project status. This concept is presented in Figure 5, Understanding and Tracking Defects and Other Metrics. This analytic process uses actual performance to re-estimate the anticipated cost and schedule. The “dashboard” at the bottom of Figure 5, presents a health and status indicator for the project. In Figure 5, five metrics are tracked, schedule variance, time variance, cost variance, size growth, and defects discovery and removal. SEER Control allows you to track size growth and actual defect metrics. Size growth can indicate growth in requirements and can be an indicator of why a project may be off track. The profile of defects reported and removed is compared against the estimated time phased defects.

In Defects Tracking, the analyst will see the estimated defects...
Understanding & Tracking Defects And Other Metrics

Heath and Status Indicator shows status and trends from the previous snapshot
- Including Size Growth and Defect Discovery/Removal Rate
- User defined control limits to control the transition between red-yellow-green

![Figure 5: Understanding and Tracking Defects and Other Metrics](Image)

reported and actual defects reported. When reported defects are lagging the estimated defects, it may indicate that not enough testing is being performed, especially if the actual defects removed are tracking with the estimated defects removed. Conversely, if the actual defects removed lag the estimated, but defects reported tracks with the estimated, then you may not have enough programming resources to make fixes. If actual defects reported and removed follow the general profile of the estimated, but are higher or lower, then the baseline project estimate may be over or underestimated. SEER Control also tracks the Time Variance (TV). The TV is the cumulative difference in schedule months between earned value and the baseline plan up to the date of the latest snapshot. When a rollup element is selected, the time variance is equal to the worst time variance of its subordinate programs. Positive values are favorable, negative values are unfavorable. SEER Control also tracks the Time Performance Index (TPI). The TPI is the time efficiency achieved from the beginning of development to the date of the latest snapshot. A performance index greater than one is favorable. A performance index less than one is unfavorable. The Time Performance Index (TPI) is the ratio of the elapsed time from the Actual Start Date to the baseline planned date and the elapsed time from the Actual Start Date to the snapshot date.

Other metrics can be tracked. In addition to the health and status indicator using the red, yellow, green indicators, this automated application re-baselines the program estimate to present a revised cost and schedule prediction.

At the heart of the PPMC vision is the desire to forecast the final project outcome based on performance to date. One of the primary goals of PPMC is to provide adequate supporting documentation (charts and reports) to support the software project management process and to satisfy stakeholder needs.

**Conclusion**

Using earned value to plan and manage software intensive projects can prevent expensive failures. Earned value should be based on the foundation of establishing the requirements, developing a reliable baseline estimate for cost and schedule, selecting effective software metrics, applying Performance-Based
Earned Value (PBEV), and using analytic processes to project cost and schedule based on actual performance.

About the Authors

Bob Hunt is the Vice President, Services of Galorath Incorporated. As Vice President for Services, Mr. Hunt is responsible for the management and technical direction of the services staff and the quality of the services products. Galorath’s professional services staff is defined by expert analysts and consultants with broad experience in cost estimating and project management, as well as deep knowledge of the capabilities, features and limitations of SEER tools. Mr. Hunt has provided software program assessments, SEI Checklist evaluations, software sizing analyses, and software cost estimating for commercial and federal clients including the Customs and Border Patrol, the Department of Defense, NASA, and various commercial clients. Prior to joining Galorath, Mr. Hunt was President of CALIBRE Services, Inc., a subsidiary of CALIBRE Systems, Inc. Prior to joining CALIBRE, he was a Vice President of Science Applications International Corporation (SAIC) responsible for the Cost and Acquisition Management Operation. As a civil servant, Mr. Hunt was Deputy Director of Cost Analysis for Automation and Modeling, Cost Analysis Division, U.S. Army, The Pentagon. In this position, Mr. Hunt was instrumental in setting up and developing the U.S. Army Cost and Economic Analysis Center and was the principal author of the initial Army Cost Analysis Manual. Mr. Hunt has held leadership positions and made technical presentations for the American Institute of Aeronautics and Astronautics (AIAA), the Society of Cost Estimating and Analysis (SCEA), and the National Association of Environmental Professionals (NAEP), and the IEEE.

Paul Solomon, PMP, is the co-author of the book, Performance-Based Earned Value*. He is internationally recognized as a leader, teacher, and consultant on Earned Value Management (EVM). He published many articles on EVM, systems engineering, software engineering, and risk management. Most are available on his website, www.PB-EV.com. He retired from Northrop Grumman Corporation where he led the use of EVM on programs including the B-2 Stealth Bomber, Global Hawk, and F-35 Joint Strike Fighter. He has taught thousands of professionals and led EVMS implementation, compliance reviews, Integrated Baseline Reviews, independent assessment reviews, and process improvement teams. He is qualified to lead EVMS certification reviews.

Daniel D. Galorath has over three decades in the software industry. Daniel D. Galorath has been solving a variety of management, costing, systems, and software problems for both information technology and embedded systems. He has performed all aspects of software development and software management. One of his strengths has been reorganizing troubled software projects, assessing their progress applying methodology and plans for completion and estimated cost to complete. He has personally managed some of these projects to successful completion. He has created and implemented software management policies, and reorganized (as well as designed and managed) development projects. His company, Galorath Incorporated, has developed the SEER applications, methods, and training for 1) software, 2) hardware, electronics & Systems, 3) Information Technology, and 4) Manufacturing cost, schedule, risk analysis, and management decision support. He is one of the principal developers of the SEER-SEM™ software evaluation model. His teaching experience includes development and presentation of courses in Software Cost, Schedule, and Risk Analysis; Software Management; Software Engineering; to name a few. Mr. Galorath is a sought after speaker. Among Mr. Galorath’s published works are papers encompassing software cost modeling, testing theory, software life cycle error prediction and reduction, and software and systems requirements definition. Mr. Galorath was named winner of the 2001 International Society of Parametric Analysts (ISPA) Freiman Award, lifetime achievement award, awarded to individuals who have made outstanding contributions to the theoretical or applied aspects of parametric modeling. Mr Galorath’s book “Software Sizing, Estimation, and Risk Management” was published March 2006. Mr. Galorath publishes a blog, “Dan Galorath on Estimating” at http://www.galorath.com/wp/.

Author Contact Information

Email: Bob Hunt [bhunt@Galorath.com]
Email: Paul Solomon [paul.solomon@pb-ev.com]
Email: Dan Galorath [Galorath@galorath.com]
Technical Training On Demand

Cost effective way for organizations to provide continuous learning opportunities for their employees

- Pay one annual subscription fee ($475)
- Take as many classes as you want

AFFORDABLE

Classes available 24/7/52
From:
- Home
- Office
- Travel

ACCESSIBLE

DACS On-Line Learning Center

COMPREHENSIVE

450+ classes
12,000 topics
Programming + Web Development Course Catalog
Latest Technologies
Java XML
Oracle Server Technologies
.NET Framework

To view the catalog visit
www.thedacs.com/training
For details call 1.800.214.7921

FLEXIBLE

- Fit course work into your schedule
- Self paced
- Refresh knowledge
- Study during lunch or after work

ACCREDITATION CERTIFICATION

ORACLE
Microsoft Certification Suite
- MCAD
- MCDI
- MCTP
- MCPD
- MCSD
- MCSE
- MCTS
- MCAS

- CEUs granted
- IACET Accreditation Supported
There are three key points in this article:

- Agile and earned value management (EVM) are a natural fit for each other
- EVM implementations can be radically simplified for agile projects
- The ideas here are relevant to all EVM users, even if you have to use ANSI/EIA-748 and/or you don’t use agile. In particular, they offer a way to share EVM results with stakeholders who don’t know any EVM jargon.

Scope Management in Agile Projects

We need to begin by busting a few myths about agile:

Myth 1: Agile development is “Extreme Programming (XP) + Scrum”.

In fact, there are many different varieties of agile development. The agile movement (formally) began in 2001 when 17 industry leaders met to identify the common ground that united their various viewpoints. All had their own approaches to building software. Some used Extreme Programming (XP), some used Scrum, some used Dynamic Systems Development Method (DSDM), some used Crystal Clear, some used Feature Driven Development (FDD) and the others used a variety of other approaches. Their intention was not to standardize their efforts on a single approach, but rather to document some common values and principles which they all agreed with. They documented those values in the Agile Manifesto, which became the “founding document” of the agile software movement. [1] The group of 17 fully intended to continue practicing their own individual approaches, under the agile “umbrella”. However in the years that followed the IT industry focused its attention on just two varieties of agile, XP and Scrum, to the extent that most people began to equate agile only with XP and Scrum.

Myth 2: Agile projects can’t define scope up front.

They can. For instance, the FDD variety of agile has always defined the full scope up front. The Crystal Clear variety has always given the option of defining scope up front. Scrum and XP are generally seen as the least supportive of up-front scoping, but even Scrum does record the functionality to be developed in an evolving list called the “Product Backlog”. Some Scrum teams create the list up front; others create it as they go.

In summary, agile processes give you the option of producing a reasonably complete definition of scope at the start of the project. That’s exactly what we need for EVM – we need overall targets for scope, time and cost so that we can track our progress towards them. You don’t have to set these targets if you just want to do agile; but you do have to set them if you want to do agile with EVM.

Agile Burn Charts

Once you have set a target for the overall scope of the project, you can track your progress towards it. Most agile teams do so with a simple chart like Figure 1, on the next page.

Agile teams typically call these “burn charts”, and they use the word “velocity” to refer to the slope of the line. Some teams chart the work that has been completed, as I have done here. Others chart the work remaining, in which case the line slopes downwards and the chart is called a burndown chart. I will
stick with upward-sloping charts in this article, because they correspond more closely to the typical EVM chart.

As I will show below, the line of an agile burn chart is equivalent to EVM’s Budgeted Cost of Work Performed (BCWP) also known as the “Earned Value” (EV). Therefore, most agile teams are already tracking earned value – although they may not use that word for it1.

The Agile Earned Value Technique

Where I work, at Optimation Ltd, we found that the simple agile burn chart was not enough for our agile projects. We wanted to understand the financial context of the team’s progress. Was the project on track? Were we making enough progress, but only by blowing the budget with overtime? If we were running late, was it due to problems in the project, or was it due to other projects “borrowing” our people?

We answered these questions by enhancing our burn charts to include the cost of development. We soon discovered that our technique closely corresponded to traditional Earned Value Management, albeit in a much simpler form.

This is how we describe it to people for the first time:

We keep track of our projects by drawing a chart with three lines on it.

The grey line shows the progress that we expect to make, as a percentage of the total project. It starts at zero and slopes up to 100% at the end of the project.

The green line shows how much of the product we have built, as a percentage of the total product size. Ideally, it should follow the same path as the grey line – starting at zero and reaching 100% completed on the project’s scheduled end date. If it falls below the grey line, that means we’re running late; if it’s above the grey line it means we’re early.

The red line shows the cost we have incurred so far, as a percentage of the total project budget. In a perfect world, it would also follow the same path as the grey line - starting at zero and finishing at 100% of budget, on the project’s scheduled completion date.

If the red line is above the green line, that means we’re spending the budget faster than we’re building the software – which is bad. If we carry on like that, we’ll have a cost overrun. Conversely, if the green line is above the red line, that means we’re building the software faster than we’re spending the money – which is good. If we carry on like that, we’ll finish under budget.

We don’t use EVM acronyms. This simple approach makes it easy to communicate the ideas to absolutely everyone involved in the project – from the most junior worker up to our most senior executives. Everyone understands the idea after just a few moments of explanation.

We don’t tell them that the grey line is Planned Value (aka PV or BCWS); the green line is Earned Value (aka EV or BCWP) and the red line is Actual Cost (aka AC or ACWP). Everyone simply refers to the lines by the names of their colors. For instance:

“The green line is still too low” = an unfavorable schedule variance (i.e. SPI < 1)

“The red line is above the green” = an unfavorable cost variance (i.e. CPI < 1)

As you may have noticed, we measure everything as percentages, rather than the usual EVM practice of measuring everything in dollars. I believe the percentage-based approach is easier for newcomers to understand: most business and technical people understand percentages with ease, but are not necessarily familiar with EVM’s notion of measuring progress as the sum-of-the-budgeted-costs-of-completed-tasks. You can convert between percentage and dollar measures simply by multiplying or dividing by the total project budget2.

---

1In fact, when agile teams say “value” they usually mean “business value”, which is the usefulness of proposed features. But in EVM “value” is not about usefulness; it is simply about size. It is the amount of budgeted cost which is considered “done” (i.e. earned) when the feature is complete. If you are an agilist reading this, then all you need to remember is that EVM “value” is the same thing as your “points”.

2This is exactly the same conversion as that shown in the Defense Acquisition University’s “Gold Card” of EVM acronyms, under the heading “Overall Status”. We just use the formula round the other way – charting the percentages and deriving the dollars.[2]
How can you use this?

Since you’ll be reading this in Software Tech News, I assume you’re working in a DoD environment, and therefore have to follow the ANSI/EIA-748 standard. The simple approach in this article can summarize the results of your ANSI/EIA-748 work, in a form that can be easily understood by all your project team, and all your external stakeholders, without requiring training in EVM. The graphical form, free of acronyms, requires no explanation other than the brief paragraphs above.

You might also find this technique useful on projects which are too small for ANSI/EIA-748. For projects worth less than $20 million, the standards-based EVM approach is not mandatory and is often omitted, due to its cost of implementation. The simpler technique in this article is cost effective on even the smallest projects. We have used it on projects worth less than $100 thousand, and here’s how we do it:

Implementing the Simplified EVM Approach

1. We simplify the distribution of Planned Value over time. We can do this because agile projects are designed to have a linear output over time – producing roughly the same amount of output in each iteration; therefore, we don’t need to compute an s-curve to draw our grey (PV) line. We just draw it with a ruler (or the computerized equivalent). This technique is not perfect, but it is very easy to implement, fits well with agile projects, and gives a result that is adequate to our needs.

2. The chart is our sole format for reporting earned value; so we do not provide numerical tables. The chart covers all current work in the project; so we don’t prepare breakdowns by WBS areas. When the chart identifies a problem, such as a dip in the green line, we use other techniques to identify the cause. (In fact, thanks to the openness and feedback of agile projects, we’re often aware of the issue before the chart confirms its adverse impact.)

3. We simplify the gathering of actual costs. In particular, we don’t break actual costs down by task. (We only graph the aggregate anyway, so we wouldn’t use the breakdown even if we captured it.) Since we’re building software, our costs are almost entirely composed of labor costs. Each week we simply record the number of hours worked on the project, during that week, by each team member. Then we sum the hours and multiply by our hourly rate, to get the cost for the week.3

4. We don’t interface to our financial or timesheet system. To save on integration and setup costs, each team member simply reads their total hours for the week out of the timesheet system, and emails (!) the number to the project manager. This sounds crude, but it is has been very effective and much cheaper than actually hooking into the timesheet system programmatically. It works because we don’t need task-level costs, only the hours per week.

5. We have one very simple “earning rule”: the earned value for a task accrues when the task is completed. You get no points for partial completion of a task. This practice is consistent with that normally used on agile burn charts and is referred to as the 0/100 rule in EVM. Furthermore, as in most agile teams, only working software features earn value. There is no earned value associated simply with designing something – you only score points when it is designed, built and tested.

6. The chart works best when the green (EV) line is “live”. It may be in a web-based or desktop tool. The key point is that changes happen straight away; when developers complete tasks, they immediately see the green line move up a little. This instant positive feedback boosts motivation and ensures that everyone cares about “moving the green line up”.

7. The chart works best when it covers a period of time that is big enough to feel like the “big picture”, but still small enough so that the team can see the line move up during the course of a (successful) day. For instance, I prefer not to use it on one-or-two week iterations since it doesn’t show enough of the big picture at that timescale. Instead, I like it to cover roughly 3 to 6 months – either the whole set of iterations that make up one “release”, or even the whole project. If your project is so big that daily changes are imperceptible, you can use two charts – one covering a period that is short enough for live/daily updates to be noticeable; and the other covering the whole project.

Continued on page 24

3 Multiply by the rate first, then sum, if different people are charged at different rates.
The DACS Gold Practice Initiative:

- Promotes effective selection/use of software acquisition & development practices
- Defines essential activities/benefits of each practice
- Considers the environment in which each practice is used
- Addresses the timeliness of practice benefits
- Recognizes interrelationships between practices that influence success or failure
- Contains quantitative and qualitative information
- Includes discussion forums, case studies and survey opportunities
- A continually evolving resource for the DoD, Government, Industry and Academia
- Free to use/free to join

Current Gold Practices:

- Acquisition Process Improvement
- Architecture-First Approach
- Assess Reuse Risks and Costs
- Binary Quality Gates at the Inch-Pebble Level
- Commercial Specifications and Standards/Open Systems
- Ensure interoperability
- Formal inspections
- Formal Risk Management
- Goal-Question-Metric Approach
- Integrated Product and Process Development
- Model-Based Testing
- Plan for Technology Insertion
- Requirements Management
- Requirements Trade-Off/Negotiations
- Statistical Process Control
- Track Earned Value

Learn More About the DACS Gold Practice Initiative:
http://www.goldpractices.com

Become a Registered Member!
http://www.goldpractices.com/registration/register_step1.php

Data & Analysis Center for Software
P.O.Box 1400
Rome, NY 13442-1400
http://iac.dtic.mil/dacs
If your environment does not support these ideas, you can still consider making percentage-based graphs for easy communication with stakeholders. Simply feed your EVM data, including non-linear Planned Value, into one simple chart, color-code it, and then discuss it by color instead of by acronym.

**Deriving other EVM Metrics – Visually**

If you can make your Planned Value approximately linear, you will reap some additional benefits: linear PV makes it easy for stakeholders to visually “reason” about the graph. People can visually extrapolate the red and green lines, to intuitively grasp many of the concepts that are difficult to learn in classic EVM. For instance, we can see this agile project is likely to overrun its budget by about 30%. (In classic EVM terms, the Estimate at Completion (EAC) is approximately 1.3 times the project budget.)

Finally, in this instance, we can see that the project is on track to complete 2 weeks early and that only about 85% of the total budget will be used when we get there.

---

**How Detailed Does the Scope Definition Need to Be?**

The scope definition doesn’t need to be any more detailed than it would be on any other agile project. For instance, a Scrum Product Backlog is fine, as is an FDD Feature List. Details can (and should) be fleshed out on an iteration-by-iteration basis in the usual agile manner. This is remarkably similar to the “Rolling Wave” approach which is described in the DoD’s Earned Value Management Implementation Guide [3], as an option available to all contractors:

2.5.2.4.2 Rolling Wave Planning. The contractor may also elect to plan the PMB in detail WPs for near term activities and hold the future budget in higher level PPs. The contractor should periodically plan the next increment of work into detailed WPs. … Government approval of or interference with the process of rolling wave planning is not appropriate; however, the CMO and PMO should be aware of the contractor’s schedule for rolling wave planning.

**What about Changes in the Scope?**

It helps to distinguish between two kinds of “changes”. The first is simply fleshing out of detail, consistent with rolling wave planning and normal agile practices. As you do so, you will find 5 Classic EVM measures schedule variance in dollars, which is counter-intuitive to the uninitiated.
unforeseen nuances of the previously-identified requirements. These are what author Robert Glass calls “derived requirements” — essential to the implementation but not explicit in the original user-facing requirements [4]. My preferred approach is to simply implement these as they arise, without reflecting them at all in the EVM system. This works because, by promptly implementing the derived requirements, you distribute them roughly evenly throughout the project. So, for example, if you have built half of the initially-known requirements (and you’ve implemented all their derived requirements as you worked); then you can be reasonably confident that you have done half of the total project. This approach minimizes the cost overhead of managing derived requirements. It also takes the pressure off the initial scoping phase, because it means the results don’t have to be perfect. They only need to be as good as the typical Scrum Product Backlog or FDD Feature List. Adding simplified EVM to an agile project does not require a higher standard of initial scoping.

The second kind of change is when the customer comes up with completely new ideas or completely new areas of functionality. These are indeed additions to the scope. The changes can be charted right on the simplified Earned Value chart. Details are beyond the scope of this article, but I have posted several examples on my web site. [5]

**Applicability**

The applicability of this technique depends on two things:

1. **Can you define scope up front?** Your definition doesn’t have to be any better than you would normally do on an agile project; but you do have to do it at the start. If you have a very exploratory project, in which you have no plans at all beyond the next few weeks, then you can’t have specific targets for the end of the project, and so you have no context to assess your earned value. You still can (and probably should) use an agile process, but you can’t use EVM.

2. **Can you achieve (roughly) linear delivery of value?** Most agile processes are designed to do exactly this. But difficulties can arise if, for instance, you lack adequate automated test coverage, so you need a large manual test phase at the end. Big changes in team size can also complicate matters. Even so, you can still tolerate some non-linearity while charting your grey (PV) line as a straight line — as long as all stakeholders understand that the grey line is a linear approximation to the plan, rather than an exact representation of it.

You might be wondering how you can apply these ideas in a regulatory environment that is governed by ANSI/EIA-748. From where I sit (8000 miles away on the far side of the Pacific) I can’t answer that — but you can. You know what’s happening in your organization, so you’re in the best position to identify improvements. Pick something which you think will be beneficial, which is achievable, and which fits within the rules. (Don’t make the mistake of limiting yourself to the most common interpretation of the rules; there may be better ways to comply.) Try the idea out on a small scale, perhaps on one of your smaller projects. If it works, use its success to justify wider usage; if it fails, test your next idea. This approach is inspired by the emerging field of Evidence Based Management. I hear that the DoD already applies Evidence Based Management very successfully in some quarters. I wish you equal success and encourage you to seek advice from within the DoD, from agile experts, and particularly from the Evidence Based Management community.

**Other Alternatives**

This technique is by no means the only way to apply EVM to agile projects. For instance, Tamara Sulaiman and a team at SolutionsIQ implemented a version of “Agile EVM” which I see as being somewhat closer to the classical EVM approach [7]. Garry Booker of Project Frontier is developing a scalable model of Earned Value Management [8]. It begins with the simplest approaches of all — such as agile burn charts which track only the completion of features. From there it can scale up to simple “3 line” methods such as the one presented in this article; then scale up a little further to the Sulaiman/SolutionsIQ approach; and finally right up to ANSI/EIA-748.

**Evaluation of the Simplified Approach**

Let’s compare the simplified implementation to the goals listed in section 1.2 of the DoD’s Earned Value Management Implementation Guide [3].

---

6 Make sure you only implement them if they really are essential to the success of the planned scope. Also, make sure you leave some spare capacity when you plan each iteration, because you’ll need it to implement the derived requirements.

7 And you have to estimate the size of each element of scope up front (using “points”, hours of effort or whatever your chosen sizing measure may be). Percentage complete (EV) is based on these sizes.

8 There is a new practice emerging in some parts of the agile community in which features are built at a minimalist level first, and then fleshed out in later iterations as/when time and budget allow. One variation is called “feature thinning” and another is called the “A-B” split [6]. They are very powerful and useful techniques, but they do make it much more difficult to do burn charts and EVM. I will not discuss them further in this article.
### DoD Goal | Simplified Implementation
--- | ---
**Relates time-phased budgets to specific contract tasks and/or statements of work (SOW)** | The “specific contract tasks” relate to the up-front scope definition. Actually scheduling those tasks, by slotting them into particular iterations, should follow your chosen agile methodology. With most agile approaches, the result will be an approximately linear flow of value.

**Objectively measures work progress** | The simplified earning rule encourages objectivity: the only thing that counts is completion of running, tested features. The insistence on the feature being tested before accruing EV is particularly important – EV is only earned when testing proves that the feature really is finished.

**Properly relates cost, schedule, and technical accomplishment** | The simplified approach relates cost, schedule and technical accomplishment in the same way as classical EVM – albeit with a linear schedule rather than the classic s-curve.

**Allows for informed decision making and corrective action** | This has definitely been our experience when using this technique – it identifies problems and allows us to respond promptly. The ease of understanding is also important. We circulate the chart to a wide audience, including decision makers who wouldn’t have the time (or training) to interpret the numbers and acronyms of classic EVM. They can, and do, correctly interpret the color-coded chart.

**Is valid, timely, and able to be audited** | Timeliness is achieved by live updates to the green (EV) line and by weekly updates to the red (AC) line. Validity is ensured by driving the green line directly out of the project’s task-tracking system, and by basing the red line on timesheet data (admittedly with a manual step to collect it).

**Allows for statistical estimation of future costs** | Future costs are estimated by linear extrapolation from the recent past.

**Supplies managers at all levels with status information at the appropriate level** | The format works well for high-level managers. For day-to-day management, project managers will naturally want more detail than what is shown in the chart. They don’t obtain that information from the simplified EVM system. Instead, they obtain it from the project’s chosen tool for agile project management / task tracking (just as they would have done if they were doing agile without simplified EVM).

**Is derived from the same EVM system used by the contractor to manage the contract** | In our case, we *are* the contractor. (We give the chart to the customer in our weekly reports.)

### Conclusion
Simplified EVM approaches are relevant to both the agile world and the classic EVM world. In fact, agile thought-leader Alistair Cockburn has described such techniques as a bridge between the two worlds.
Acknowledgements: Thanks to Garry Booker, Alistair Cockburn and Bob Sutton.

References


[6] For an overview of Feature Thinning and related techniques, see http://alistair.cockburn.us/The+A-B+work+split,+feature+thinning+and+fractal+walking+skel etons


About the Author

John Rusk is a software architect at Optimation Ltd in Wellington, New Zealand. He is an active member of the agile development community, and the author of two open-source software projects. He writes about agile software development at www.agileKiwi.com. He has worked in the Wellington software industry since 1995, creating bespoke software for private and public sector organizations. He has a B.Sc. in Computer Science from Massey University, New Zealand.

Author Contact Information

Email: John Rusk [john@agileKiwi.com]

---

Key points for agile teams

- You can, and often should, examine scope up front. For instance, you may populate your “Product Backlog” with an initial approximation of the full feature set.
- You can, and often should, add “budget burn” (AC) to your burn chart. Doing so gives a simple, but effective, earned value chart that conveys considerably more information than a standard burn chart.

Key points for users of classic EVM

- When presenting EVM to stakeholders who aren’t project managers, consider presenting it graphically, without acronyms.
- Simplified implementations don’t need per-task actual costs.
- Simplified implementations can use linear Planned Value if they use agile techniques to achieve approximately linear output.
- Showing “live” changes to EV is good. It boosts motivation if the team can see their day’s work having a direct impact on the green line.
- Use Evidence Based Management to improve your EVM implementation: what is working well? What is working poorly? Which changes will you try?
Are you getting **All** you can from your software investment?

The DACS ROI Dashboard

**Impact on Quality (% defect reduction)**

- 75th Percentile: 90
- Oklahoma City Air Logistics Center, Test Program Set and Industrial Automation line observed the defect density decrease 91%, from 3.3 to 0.3 Defects per KSLOC, while achieving CMM Level 4.

**Fact Information**

- CMM Software Process Improver
- Fact: Improved
- From 1999 to 1999, Oklahoma City Air Logistics Center, Test Program Set and Industrial Automation line observed the following improvements:
  - CMM Software Process Improvement
  - Improved
  - From Level 2 to Level 3 in 1999
  - Improved
  - From Level 4 to Level 5 in 1999

**Source:**


Access the DACS ROI Dashboard!

http://www.thedacs.com/databases/roi/

**Technologies Covered:**

- SEI CMM/CMMI
- SEI Team Software Process (TSP)
- SEI Personal Software Process (PSP)
- Inspections
- Reuse
- Cleanroom

**And Many More!**

**Graphs Showing Impact Of Software Technologies On:**

- ROI
- Productivity
- Quality

**Summarizes Facts From Open Literature**

Data & Analysis Center for Software
P.O. Box 1400
Rome, NY 13442-1400

http://iac.dtic.mil/dacs
Defense Acquisition University’s Integrated Program Management Model

We know that for a program that is behind schedule, the schedule variance percent metric is highly correlated to the critical path.

by David C. Bachman, Defense Acquisition University

The Defense Acquisition University’s (DAU) Integrated Program Management Model (12-step model), shown in Figure 1, is a work in progress originally created in 2000 to answer an Air Force Lieutenant General’s question: How do you use earned value to manage? The first version of the model had ten steps. Contract analysis (Step 3) and financial analysis (Step 11) were added in 2001. The AT&L Trip Wire metrics were added to the model in 2008. The model provides program managers with a disciplined sequential approach to evaluating scope, schedule, and budget data needed to determine current program status. The results of the analysis become the foundation for oversight reporting to higher levels of management. We’ll explore the answer to the General’s question as we examine the 12 steps. Note that for the remainder of this article, references to items represented as objects in the Model diagram will appear in italics.
The 12-step model assumes that an earned value management (EVM) performance measurement baseline (PMB) exists and that mandatory EVM reports are delivered on a recurring basis from the contractor. The 12-step model analysis begins after the completion of the EVM integrated baseline review (IBR). The April 2003, Program Manager’s Guide to the Integrated Baseline Review Process, states: “The intent of the IBR is to provide the PMs with a mutual understanding of the project PMB and to attain agreement on a plan of action to handle the identified risks.” The documented results of the IBR are an EVM PMB, an integrated master schedule, and an updated risk management plan which brings us to the starting point of the 12-step model.

Step 1 – Data Collection - The 12-step model requires the collection of key program and performance data from a variety of sources. The 12-step model’s first required document is the updated risk management plan (Risk Mangt Plan) that incorporates results from the IBR. The second is the program manager’s acquisition program baseline (APB). This document represents a contract between the program manager and the milestone decision authority and incorporates the warfighter’s requirements. The next document is the “Contract” which establishes the scope, schedule, budget, and technical performance expected from the contractor. The next three documents are standard contract data deliverables prepared by the contractor: a Network Schedule in accordance with the integrated master schedule (IMS) data item description (DI-MGMT-81650); all five formats of the contract performance report (CPR) (DI-MGMT-81466A); and the contract funds status report (CFSR) (DI-MGMT-81468). The next two groups of documents are related to program funding:

- R & P Forms are associated with the Department of Defense’s (DoD) Program, Planning, Budgeting, and Execution (PPBE) system. These documents are prepared and maintained by the program office’s PPBE process. They are included as part of the President’s Budget and ultimately approved by Congress.
- DFAS O&E documents are budget obligations and expenditures reports from the Defense Finance and Accounting Service.

Once the program has been funded through the PPBE process, financial execution is tracked by DFAS O&E. The final data item deals with the earned value management system (EVMS) surveillance. The contract administration office is responsible for routine on-going EVMS surveillance (DCMA Surveillance). In the DoD, routine surveillance is accomplished by either the Defense Contract Management Agency (DCMA) or the Supervisor of Ship Building (SUPSHIP). DoD policy requires that the results and findings of this routine EVMS surveillance be documented and available for review by program managers.

Step 2 – Acquisition Program Baseline (APB) Analysis

Step 2 of the 12-step model is undertaken to identify and understand the performance, cost, and schedule objectives against which the program manager is measured. The APB establishes parameters for technical maturity, affordability, and programmatic requirements that must be demonstrated at a milestone review to progress to the next acquisition phase. The APB defines thresholds and objectives for the program’s technical performance (Performance KPPs) and schedule (Schedule Parameters). It also establishes cost and cost as independent variable (CAIV) objectives (Cost Parameters). The APB identifies the warfighter’s key technical performance parameters (KPP). Understanding the APB threshold and objective requirements provides the program manager with trade space needed to manage the program. Although the 12-step model focuses the analysis effort on the KPP requirements, it is also important to note that program managers are responsible for achieving all APB requirements, not just the KPP requirements.

Step 3 – Contract Analysis - With an understanding of program requirements from the APB in step two, step three maps the program requirements to the contract requirements. Ideally the contract requirements should be identical to the APB requirements, but in today’s spiral development environment, they are often different. In addition to the original contract, step three includes four critical data elements. The first is the contract modification status called P0000 Status. P0000 mods are initiated by the Procuring Contracting Officer (PCO), but in this model, P0000 Status represents all contract modifications or pending contract modifications whether they are initiated by the PCO or the Administrative Contracting Officer (ACO). The Contractor Finances block requires determining the contractor’s financial status and the current contract funding levels. SPEC Req refers to the contract’s specification requirements and are reviewed to identify the contract’s technical requirements. The Delivery Req box refers to the negotiated delivery dates written into the contract. Documenting and understanding any differences that
may exist between the contract documents (Step 3) and the APB requirements (Step 2) is essential to determining potential contract issues in later steps of the 12-step model.

**Step 4 - Technical & Risk Analysis** – Step 4 in the 12-step model incorporates the system engineering process and results in the first three analysis decisions (represented by diamonds in the 12-step model). During the IBR, we identified the technical, schedule, resource, cost, and management risks which resulted in an updated risk management plan (Risk Mangt Plan). This plan, the APB technical “Performance KPPs” and the contract specifications (SPEC Req) provide the basis for step four's technical and risk analyses. The first elements of Step 4 are the development and updating of the program Risk Matrix and technical performance metrics (TPM). These items are designed to measure technical progress and to manage program risks. The TPMs and the Risk Matrix may actually be tracked and maintained by the contractor as part of the program's system engineering process. Once defined, these metrics need to be maintained, updated, and statused on a recurring basis. Using this information, we make the first 12-step model decision by determining the technical and risk drivers (Tech & Risk Drivers). Confirmation that the warfighter’s Performance KPPs are still achievable is the second step 4 decision (Status Performance KPP). A decision that the Performance KPPs are not achievable would require an APB revision. In the final Step 4 decision, we’ll confirm that the contract’s technical requirements (Status SPEC) are still feasible. An unachievable contract specification requires a contract modification. Early identification of technical issues is essential to integrated program management. Step 4 is intended to identify the program's technical drivers, and to evaluate whether or not the APB performance requirements and the contract’s specifications are achievable. Because these metrics are dynamic, these analysis decisions may change as the contract evolves.

**Step 5 – Data Validity Check** – Step 5 of the 12-step model requires a decision that the Network Schedule, the CPR, and the contractor’s EVMS are providing good data that actually represents the current program status. We’ll revisit Step 5 when we get to Step 11 for the financial management documents. The Network Schedule and CPR are normally required data deliverables when using EVM. In Step 5, we assess the CPR validity; the IMS schedule health metrics, the AT&L System Indicating Trip Wire1 metric and the AT&L Baseline Indicator Trip Wire1 metric. If the Network Schedule and/or CPR are contrived or incomplete, we would be wasting our time analyzing them in Steps 6 through Step 10. Likewise, we should question the validity of any report generated by a flawed EVMS. Surveillance of the contractor’s EVM system is the responsibility of the contract administration office; typically this is the DCMA, but it could also be SUPSHIP at shipyards. These organizations perform and prepare recurring reports (DCMA Surveillance) on two levels of surveillance: EVM System surveillance; and program surveillance. Both reports should be reviewed and assessed as part of the Step 5 data validity check.

**Step 6 – Network Schedule Analysis** – Step 6 is where we begin to answer the General’s question of: How do you use earned value to manage? With valid data from step 5 we can now evaluate the program’s Network Schedule, also known as the integrated master schedule (IMS). It is important to define two terms: baseline schedule and current schedule. The baseline schedule represents the original approved plan to accomplish the program’s objectives. The baseline schedule is established at the IBR. The current schedule is an updated baseline schedule which reflects actual accomplishments to date and projections for completing the remaining objectives. Step 6 begins by calculating the AT&L schedule performance Hit / Miss Index metric1, AT&L Baseline Execution Index (BEI) Trip Wire metric1, and AT&L Critical Path Length Index Trip Wire (CPLI) metric1. It continues by evaluating the current schedule’s critical path”, milestone completion dates, and by determining the schedule status of the technical and risk drivers (Tech & Risk Drivers) identified in step four. When required by contract, Step 6 may include a Monte Carlo type “what if” schedule risk assessment called SRA in the model. The SRA provides potential finished dates based on risk assumptions. Step 6 results in two decisions. The Status Schedule APB decision compares the IMS milestone completion dates and with the APB’s schedule parameters from Step 2. The Status Delivery decision evaluates the IMS critical path schedule to ensure that the Step 3 Delivery Req contractual delivery dates are realistic and achievable. Step 6 continues by analyzing the Float, also called slack time, for the Step 4 Tech & Risk Drivers and by reviewing slipping tasks. This is the beginning of how to use EVM to manage. We would expect that the technical and risk drivers would correlate to a great extent with the critical path. Not all items identified in Step 4 will appear on the critical path, but any items on the critical path that have not been identified as “drivers” need to be evaluated to determine if they belong on the list.
Step 7 – Performance Measurement Baseline (PMB) Analysis - The final activity identified in the Program Manager’s Guide to the Integrated Baseline Review Process is management processes. The guide lists three specific management processes: 1) the risk management process; 2) business processes in general; and 3) the baseline maintenance process. In Step 4 of this model, we addressed the risk management process. The model itself deals with the business processes of scheduling, estimate to complete, earned value methodology, and managerial analysis. In this step, we’ll specifically address the baseline maintenance process by ensuring that the PMB represents the current plan for accomplishing the remaining work, and that, it also incorporates any changes resulting from program dynamics. Step 7 begins by evaluating the program’s Contract Performance Report (CPR) format three (Baseline), format four (Staffing) and format five (Explanations and Problem Analyses) (CPR Format 3, 4, & 5). At the completion of the Format 3 review, the AT&L Contract Modifications Trip Wire and the AT&L Baseline Revisions Trip Wire can be determined. CPR formats three and four are forward looking. EVMS guideline 28 states:

Incorporate authorized changes in a timely manner, recording changes in the budgets and schedules. In the direct effort prior to negotiation of a change, base such revisions on the amount estimated and budgeted to the organizations.

We begin this step by confirming that the CPR PMB format three includes all negotiated contract modifications and government authorized unpriced work (Step 3 – P0000 Status); and that the CPR staffing format four includes resources for this work. The next part of step seven analyzes the amount and use of management reserve (MR); appraises the amount of undistributed budget (UB); and evaluates any single point adjustments (SPA) and/or over-target baselines (OTBs). The program manager’s report section of the CPR format five should address these PMB related changes. These assessments lead to the PMB Status decision which is the single most important EVM related decision associated with the 12-step model. This decision must affirm that the EVM performance measurement baseline correlates with the actual program status. If it does, EVM metrics can be used to assess program risks and to compute a meaningful estimate at completion range. If it doesn’t, other program management tools will need to be used to evaluate program risks.

Step 8 – Schedule Drivers - Step 8 of the 12-step model uses EVM to determine the program’s schedule drivers by correlating EVM schedule metrics with the critical path schedule and with the Technical & Risk Drivers. The CPR Format 1 (Work Breakdown Structure) reports current and cumulative EVM performance data based on the contract work breakdown structure. From this data, the schedule variance (SV) and the schedule variance percent, (SV%), EVM metrics are computed and rank ordered. Tools such as wInsight 5 and iPursuit 6 have automated these sorting and computation tasks. Once we have identified the WBS elements with the highest SV and SV%, we’ll want to confirm correlation with the Tech & Risk Drivers. Logic suggests that our high risk technical WBS elements will also be our schedule drivers. We also need to correlate the EVM schedule variance metrics with our Critical Path schedule. We know that for a program that is behind schedule, the schedule variance percent metric is highly correlated to the critical path 7. Unfavorable schedule variances result when a work package is not completed on time or when scheduled work is not started on time. A slipping critical path element typically has work packages with both of these attributes. The first Step 8 decision is, choosing the program’s Schedule Drivers. Ideally the Step 4, Technical & Risk Drivers, Step 6, Critical Path schedule, and Step 8, EVM schedule metrics will all correlate. When they don’t, the reviewer must be able to reconcile any inconsistencies. There are no absolute rules for reconciling the differences; but I recommend that you first weigh the Critical Path schedule, then the SV% metric, then the SV dollar metric, and finally the technical risk driver decisions. The second Step 8 decision is a CAIV decision. The model labels this decision as Contract Issues and it is especially important for cost reimbursement type contracts. If the contractor is addressing schedule drivers by using premium time, working multiple shifts with less skilled workers, hiring expensive consultants or any other number of costly options, CAIV requires we reevaluate the technical
and schedule requirements seeking lower associated costs. The CPR Format 5 should provide insight into program schedule drivers from the control account manager’s perspective. Cost drivers are addressed in Step 9.

**Step 9 – Cost Drivers** – Step 9 determines the program’s Cost Drivers by correlating EVM cost metrics with the program Schedule Drivers, with the Critical Path schedule, and with the Tech & Risk Drivers. From the CPR Format 1, the cost variance (CV), and the cost variance percent (CV%) are computed and sorted. With our CV and CV% metrics, we’ll first want to confirm correlation with the Tech & Risk Drivers. We would expect that our technical and risk drivers will also be our cost drivers. Next, correlate the EVM cost variance metrics with the Critical Path schedule. Finally, we correlate the cost variance metrics with the Schedule Drivers. We know from practical experience that unfavorable EVM schedule metrics are an excellent forecaster of future unfavorable EVM cost metrics. The CPR Format 5 should provide insight into program cost drivers from the control account manager’s perspective. We would like to see a high degree of correlation among the Tech & Risk Drivers, the Critical Path schedule, the Schedule Drivers, and the EVM cost variance metrics to support our program cost driver decisions. When there are inconsistencies, I recommend that you first weigh the CV dollar metrics; then the critical path schedule, then Tech & Risk Drivers, and finally the CV% metrics. Like in step eight, the second step nine decision is a cost as an independent variable decision labeled as Contract Issues. At this point in the 12-step model, we can answer the “how do we use EVM to manage” question. When the earned value cost and schedule metrics are consistent with the Step 4 Tech & Risk Drivers, and the Step 6, Critical Path schedule, we can conclude with a high degree of confidence that we understand the program risks. When they don’t correlate, we must at least understand why they don’t and consider changing our Tech & Risk Drivers decision. For instance, let’s suppose that the kill vehicle, a major element of a missile defense system, is determined to be the number one technical risk driver for a missile defense system. But our critical path schedule shows that the second stage booster, also a major element of a missile defense system, is on the critical path and slipping, and the booster’s earned value data reports a fifty percent unfavorable schedule variance, and a seventy five percent unfavorable cost variance. By comparison, the kill vehicle has ten days of schedule float and only a ten percent unfavorable cost variance and no schedule variance. With this additional integrated program management data, we may want to rethink our technical and risk decision to include the second stage booster as a technical and risk driver as well.

**Step 10 - EAC and PAC** – Step 10 of the 12-step model requires the computation of the estimate at completion (EAC), and conversion of that cost into a price at completion (PAC) that includes profit/fee. With a valid and realistic performance measurement baseline (Step 7), EVM can provide a reasonable estimate at completion range. Step 10 starts with the evaluation of CPR Formats 1, 4, & 5, and continues by evaluating contract level trends in preparation for determining the appropriate performance factors to compute an EAC range. The key EVM performance factor metrics are the schedule performance index (SPI) and the cost performance index (CPI). These are also AT&L Trip Wire metrics. In the 12 step model, the representative SPI and CPI trends should be based on the cumulative SPI for the Critical Path and Schedule Drivers. Now, if a task has any float time, an unfavorable SPI is not representative of the true status of the project. The CPI and SPI trends should be representative of the cumulative cost performance index of the PMB performance. The EAC based on the cumulative CPI performance factor generally represents the floor of the EAC range. The EAC based on the composite performance factor cumulative CPI times cumulative SPI generally represents the upper end or ceiling of the EAC range. With the determination of an EAC range, the final Step 10 tasks are to document the AT&L CPI and SPI Trip Wire metrics, to compute the AT&L to complete performance index (TCPI) Trip Wire metric, and to convert the EAC into a price at completion (PAC) by adding the appropriate fee or profit. This decision is documented in the model as (EAC PAC). After these computations, the focus of the 12-step model changes from EVM to financial management.

**Step 11 – Financial Analysis** – Step 11 of the 12-step model is intended to ensure that the program is defendable, credible, and consistent with APB requirements and the PPBE documentation. Before we can start Step 11 we must complete Step 5 by ensuring that the CFSR, the R & P Forms, and DFAS O&E are accurate and complete. The Step 11 financial analysis begins by reconciling the PPBE budget documents. The 12-step model labels these as R & P Forms. The language in these documents should match the APB language. The budgets and schedules reported in these documents should be completely consistent with the Step 10 PAC and the Step 6 Critical Path and milestone schedules. Reconciling the CPR and CFSR is the second Step 11 analysis. The CPR formats one, three, and four should be consistent with projected CFSR funding, expenditure, and termination liability requirements.
With a reconciled CFSR, the 12-step model applies Step 10 EVM performance factors to project most likely funding requirements and expenditure projections. These data are compared with DFAS obligation and expenditure reports (DFAS O&E) to make the Step 11 Budget Execution decision and to ensure the program avoids potential anti-deficiencies and misappropriations. The second financial analysis decision (Status Cost APB) is confirming that the APB’s Cost Parameters are still achievable. The final step eleven decision (Status Cont Fin) confirms that the contractor and subcontractors have adequate corporate liquidity to complete the program.

**Step 12 – IPT Reports** - Step 12 documents the results of the first 11 steps, presenting results of analyses and formatting the data for leadership and oversight reporting. The first integrated product team (IPT) report is the contract issues report. In Step 3 we reviewed the contract status, contractor finances, specification, and delivery requirements. In Steps 4, 6, 8, 9, & 11 these requirements were evaluated in the context of integrated program management. Any identified contract issues should be reported for leadership resolution. The second IPT report identifies APB Issues Trip Wire values. By combining our Contract Issues with Step 4’s APB Status Performance KPP decision, Step 6’s APB Status Schedule APB decision, and Step 11’s APB Status Cost APB decision, we can identify and report potential or actual APB breaches requiring milestone decision authority resolution. AT&L Trip Wire metrics are evaluated in steps 5, 6, 7 and 10. Although the model emphasizes KPP requirements, the program manager is responsible for achieving all APB requirements, not just the KPP requirements. The EV Status Report is the third IPT report. It essentially documents the results of Steps 7, 8, 9 & 10. Contract cost and schedule drivers are reported and the EAC most likely range is justified to leadership by this report. The Financial Status Report is the final Step 12 report. It confirms that the PPBE documents provide adequate resources to complete the program and that budget obligations and expenditure metrics are consistent with OSD target percentages. Including testing as an element of Step 4’s system engineering process, we have now collected all the data required to complete the seven sections of the quarterly defense acquisition executive summary (DAES) and the process begins again in the next reporting period. EVM data is typically reported by the contractor on a monthly basis.

**DAU Integrated Program Management Model Conclusion**

The DAU 12 -step model is a work in-progress designed to provide a structured approach for integrated program management analysis which includes EVM. A basic premise of the model is explaining how earned value management relates to integrated program management. The model shows how EVM metrics either reinforce technical, risk, and program management decisions or identify new areas requiring management attention. Additionally EVM metrics can be used to develop an estimated cost at competition range and to forecast financial execution performance data needed to avoid potential anti-deficiencies and misappropriations. DAU and this author offer this 12-step model as one way to do program analysis but not the only way. DAU has incorporated this model into the BCF-203 intermediate earned value management course and the PMT-401 program mangers course. Two and three day performance support workshops are also available from DAU on this topic. An animated presentation of the model is available at no cost on the Acquisition Community Connection’s (ACC) Earned Value Management Community of Practice (CoP). Link to the presentation:


Readers interested in learning more about the model through a real world scenario are invited to view the author's presentation available from the DACS web site at:

https://www.softwaretechnews.com/media/12_Step_Interactive_Final.ppt

**Endnotes**

1. In late 2006, the Defense Contract Management Agency (DCMA) working with the Office of the Deputy Under Secretary of Defense for Acquisition, Technology and Logistics (ODUSD(AT&L)) developed a set of metrics and thresholds designed to provided early warning of unfavorable program cost, schedule, and performance trends. These metrics provide the Under Secretary of Defense for Acquisition, Technology and Logistics (USD(AT&L)) with new management indicators needed to oversee Major Defense Acquisition Programs. These “AT&L Trip Wire metrics” were add to the DAU integrated program management model in 2008.

2. R & P Forms is a widely used Department of Defense financial management term associated with the Planning, Budgeting, and Execution (PPBE) system. R-forms are associated with Research, Development, Test, and Evaluation appropriations. P-forms are associated with Procurement appropriations.

3. DFAS O & E is a widely used Department of Defense
financial management term associated with obligations and expenditures. An obligation is binding agreement that will result in expenditure immediately or in the future. An expenditure is an actual disbursement of funds in return for goods or services. Obligation and expenditure percentages are key metrics used to measure the efficient use of budget authority.

4. P0000 Status is a widely used Department of Defense contracting term used to denote a contract change. The first contract change would be documented as P0001 the second as P0002. The P indicates that the change was accomplished by the procuring contracting officer.

5. wInsight is an EVM tool of the Deltek Corporation for analyzing, sharing, consolidating and reporting earned value management data.

6. iPursuite is an EVM analysis tool of Dekker, Ltd. that can be used to monitor EVM performance.

7. Critical Path - A sequence of discrete tasks/activities in a network schedule that has the longest total duration through the contract or project. Discrete tasks/activities along the critical path have the least amount of float/slack.

8. Float/Slack - The amount of time a task/activity or milestone can slip in a scheduled before it delays the contract or project finish date.

9. Anti-Deficiency - An anti-deficiency occurs when a contracting officer or program manager incurs obligations or expenditures (outlays) in excess of the amounts available in appropriations. This is a violation of federal law.

10. Misappropriation – A misappropriation is a Department of Defense financial management term used to describe the misuse of funding. For example, if production funding was used to fund a research and development project this would be a misappropriation. This is a violation of federal law.

His industry experience includes work as a Senior Program Analyst with Engineering Management Concepts, Inc., providing Earned Value Management support to the Missile Defense Agency. His duties included the curriculum development and teaching of twelve command sponsored continuing education courses to 500+ DoD and Industry Students.

Dave's military background includes extensive experience as a Navigator, Rated Staff Officer, JCS Intelligence Officer, and as a member of the Acquisition Corp. He started his acquisition career as an “Education with Industry” student at Hughes Aircraft Company. He was twice assigned to the GPS Joint Program Office, first as the project officer for the B-52 – GPS integration, and then as the Nuclear Detonation Detection System's Program Manager.

Dave completed his military career at the Defense Systems Management College in Fort Belvoir, Virginia, where he was the course manager for the Intermediate Earned Value Management Course. Dave has a Masters in Education from Marymount University, an MBA from Rutgers University, a BS in Ceramic Engineering from The Pennsylvania State University and he won the 2005 United States Distance Learning Association’s silver award for excellence in teaching.

About the Author

Dave Bachman is a retired military officer who has worked in industry and government. He is currently the Performance Learning Director for Earned Value Management at the Defense Acquisition University (DAU) and the editor of the Acquisition Community Connection's Earned Value Management Community of Practice. His duties include knowledge management, curriculum development, acquisition research, and consulting.

About the Author

Dave Bachman is a retired military officer who has worked in industry and government. He is currently the Performance Learning Director for Earned Value Management at the Defense Acquisition University (DAU) and the editor of the Acquisition Community Connection's Earned Value Management Community of Practice. His duties include knowledge management, curriculum development, acquisition research, and consulting.
**SPRUCE: Systems and Software Producibility Collaboration and Experimentation Environment**

THE PRIMARY OBJECTIVE OF SPRUCE IS TO ADDRESS THE TECHNOLOGY TRANSITION PROBLEM AND BRIDGE THE “VALLEY OF DISAPPOINTMENT”

by Patrick Lardieri, Rick Buskens, Srinivasa Srinivasan, Lockheed Martin Advanced Technology Laboratories, William McKeever and Steven Drager, Air Force Research Laboratory

---

**Executive Summary**

The Systems and Software Producibility Collaboration and Experimentation Environment (SPRUCE) is an open web portal to bring together DoD software developers, users, and software engineering researchers virtually by enabling their collaboration on specifying and solving software producibility challenge problems. SPRUCE is based on the premise that well articulated and bounded problems can spark scientific and engineering innovation in software producibility and help to bridge the gap between technology users and technology providers. Key SPRUCE features are: self-organizing communities of interest (CoI), dynamically evolving challenge problems with accompanying artifacts, and built-in experimentation facilities to reproduce the problems and evaluate solution benchmarks. To participate in SPRUCE, visit [www.sprucecommunity.org](http://www.sprucecommunity.org) and request an account.

**Why SPRUCE?**

Consider an engineer or architect working on a DoD program, putting out day-to-day fires and still having to consider and deal with a variety of technical problems at depth. Today, they have no easy means to explore if someone has already encountered similar problems and therefore may have some unique insights to offer. SPRUCE enables sharing of problems and insights by giving the engineer or architect a platform to articulate specific problems, using concrete artifacts and repeatable experiments. By posting the challenge problem, associated artifacts and experiments in SPRUCE, the engineer can also help future programs that could face the same problem.

Next, consider a software producibility researcher or graduate student, focused on building exciting technologies and tools. Today, there is no readily accessible repository of ‘real-world’ challenge problems and data on which they can demonstrate and validate their technologies. Furthermore, such artifacts may give researchers more insight into actual challenge problems, helping to prevent them from making incorrect assumptions. SPRUCE provides access to such a repository and also enables a perspective on how their tools and technologies may ultimately find their way into practice.

SPRUCE’s vision is to satisfy this multitude of stakeholder needs by bringing them together through (a) well-defined, in-depth challenge problems and program-representative artifacts; and (b) repeatable benchmarks and experiments that can be readily conducted in an attached experimentation facility.

Figure 1 reflects the current process within the DoD ecosystem for identifying, developing, and transitioning software producibility technology. Government personnel working DoD acquisition programs coordinate with government personnel working research programs to define software producibility problems and research agendas. The problems are then described and written into research programs’ Broad Agency Announcements (BAAs) and performers are asked to bid specific development and transition plans for software producibility solutions.

![Figure 1: Current technology identification, development, and transition process](image)

Software producibility researchers are then awarded contracts to develop their technology. Unfortunately, these researchers typically have little or no relationship with engineers in the program or domain from which their particular challenge problem is derived. While researchers strive to understand and incorporate deep, specific knowledge about a problem domain, it is hard for them to obtain detailed information and even harder when classification and International Traffic in Arms Regulations (ITAR) issues are involved. Researchers thus have little choice but to design and conduct experiments that are abstract and typically...
small-scale representations of the real challenge problem. While these results may show the promise of the new technology, they leave a large “credibility gap” in the minds of program engineers about whether the results will transition into the real problem domain. History indicates that it is hard to successfully bridge this gap, leading to the “valley of disappointment” shown in Figure 1. The ultimate success or failure of technology transition thus depends on the ad hoc, opportunistic transition process described above where serendipity of the right people being in the right positions is the primary enabler for success.

The primary objective of SPRUCE is to address the technology transition problem and bridge the “valley of disappointment” described above. SPRUCE emphasizes artifacts (e.g., sanitized DoD application software, computational resources such as specialized avionics processors and workflow management tools and services), typically provided in the context of challenge problems, and experimentation around them to create a common clearinghouse for program engineers and technology researchers to discover joint interests and form collaborations. Collaborations on real world software producibility challenges and the associated experiments using realistic artifacts are the key to successful technology transition.

What is SPRUCE?
SPRUCE is an open, collaborative research and development environment to demonstrate, evaluate, and document the ability of novel tools, methods, techniques, and run-time technologies to yield affordable and more predictable production of software-intensive systems. The key elements of SPRUCE are: (1) a collaboration environment that enables and sustains active collaboration between various stakeholders, allowing stakeholders to describe and discuss challenge problems, potential solutions, and experiments to showcase the problems and evaluate solutions, and (2) an experimentation environment containing realistic software artifacts and computing systems that promote scientifically rigorous evaluation. The following sections describe the key SPRUCE elements in more detail.

SPRUCE Collaboration Environment
The SPRUCE collaboration environment, implemented as a web portal, seeks to empower its users to define and evolve narrow, well-defined technology problems of mutual interest—but at depth—and seeks to provide them with tools for collaboration and discovery. To achieve this goal, SPRUCE structures its collaboration environment around four basic concepts: communities of interest (Col), challenge problems, candidate solutions, and experiments and experiment instances (Figure 2).

Communities of Interest (Col): Communities of interest serve to organize SPRUCE content (i.e., challenge problems, candidate solutions and associated discussions) around broad but focused topic areas. They also serve as a virtual meeting place for SPRUCE users. SPRUCE users can belong to one or more communities of interest.

Challenge Problems: SPRUCE challenge problems represent sanitized versions of realistic problems that may occur on actual DoD acquisition programs. These problems may have occurred on other DoD programs in the past, may express a desire to solve future anticipated problems that would be tedious to solve using existing means, or may provide a context for radically new approaches to systems and software development. As these challenge problems represent a shared concern, they provide an opportunity to bring together the various stakeholders in the DoD software-intensive systems producibility (SISP) ecosystem. SPRUCE encourages and enables DoD programs to submit realistic and sanitized artifacts that accompany challenge problems to attract researchers and provide real-world depth for challenge problems.

Candidate Solutions: SPRUCE candidate solutions describe proposed solutions to SPRUCE challenge problems. Since SPRUCE challenge problems represent realistic problems faced by DoD programs, successful SPRUCE candidate solutions are more amenable to technology transfer. Researchers and tool vendors may, if desired, elect to upload their technology and tools into SPRUCE and to associate licensing conditions with the use of the tools. More likely, however, SPRUCE will be used to highlight specific properties of the tools and solutions and how they address specific challenge problems posed. Researchers and tool vendors can provide links to their solutions for interested
SPRUCE users to access.

**Experiments:** SPRUCE experiments are associated with challenge problems and candidate solutions, and serve two primary purposes: (a) to showcase scenarios described in a challenge problem, so that SPRUCE community members have a repeatable baseline or (b) to evaluate the effectiveness of a particular solution or set of solutions against a benchmark. In the former case, they are best initiated and mediated by the challenge problem provider, whereas a solution provider is best suited to define and conduct the latter kinds of experiments. **Experiment instances** represent an instantiation of a SPRUCE experiment that can be run on actual hardware, including the SPRUCE experimentation environment (discussed in the next section).

As shown in Figure 2, challenge problems, candidate solutions and experiments are interrelated, and each can belong to one or more communities of interest. To facilitate a community’s access to collaboration, SPRUCE automatically creates artifact repositories, community wiki and discussion fora (termed ‘collaboration facilities’) for each of these entities and makes them readily accessible from the entity’s main page. The use of social networking tools and instant communication facilities, such as rich text and media chat, as well as member presence information is being considered for future capabilities.

**SPRUCE Experimentation Environment**

In addition to the web portal, SPRUCE provides an experimentation environment that is available to all SPRUCE users. This environment, comprised of real hardware resources, can be used to illustrate challenge problems and showcase candidate solutions in a repeatable manner on a representative environment. The SPRUCE experimentation environment is based on Emulab (www.emulab.net).

SPRUCE users can access the experimentation environment remotely, request and receive experiment resources, setup desired experiment configurations, download specific operating systems and software, conduct experiments, and collect results. This interaction is done manually with the help of scripts. Results of the experiments can be posted to SPRUCE wiki pages to be shared with other SPRUCE users. Future work will enable the seamless integration of the experimentation environment and the collaboration environment, allowing reuse, cataloging, and automation of many of these functions. The use of a community wiki to post and discuss experimental results enables community-driven peer review and discussion, much like Wikipedia®, thereby enhancing the credibility of the content.

**Examples in SPRUCE**

The SPRUCE portal (http://www.sprucecommunity.org) is currently in Beta phase with a limited number of registered users validating use cases and collaborating around an initial set of challenge problems associated with an initial set of communities of interest. Current SPRUCE communities of interest include: (1) real-time and embedded systems, (2) multi-core architectures, and (3) feature-oriented software analytics. The screen images shown in Figures 4, 5, 6 and 7 showcase initial challenge problems, experiments, and collaborations posted in SPRUCE.

Figure 4 illustrates the main page of a challenge problem that serves as the landing page for the community interested in
this problem. The main pages of other SPRUCE components are structured similarly. Each challenge problem has associated metadata: title, description, sponsors, keywords and a collection of CoIs (label 1 in Figure 4). There are dedicated collaboration facilities, such as discussion forum topics, wiki pages, and artifact repositories associated with a challenge problem (label 2 in Figure 4). Lists and hotlinks to related SPRUCE entities such as experiments, challenge problems, and candidate solutions are also available for easy navigation and cross-reference (label 3 in Figure 4).

Figure 4 shows a sample challenge problem related to “cache false-sharing” in multi-core architectures. In this problem, conflicting cache requirements of programs running on multiple processor cores constantly invalidate the processor cache and thus cause performance degradation by defeating the purpose for which the cache was designed. This challenge problem is part of the “multi-core architectures” CoI.
Figure 5 shows a wiki entry from an experiment conducted for the “cache false-sharing” problem of Figure 4. The wiki allows for free-form analysis of experiment results and generation of potential ideas for future research. SPRUCE users editing the wiki page can also create links to other places in the SPRUCE portal for easy navigation.
Figure 6 shows an artifact associated with another challenge problem in multi-dimensional resource optimization. The data set is a sanitized version of an artifact from a military and aeronautics application and represents 11,406 types of network messages flowing across 46 processors. Each message’s size, frequency, semantic type, etc., are specified as shown in the top pane. The CPU loading for 14 of the processors, which are available for allocation, are also specified. Collaborations are currently in progress to develop further assumptions, experiments, and solutions to this challenge problem. Since the challenge problem provider also serves as an active moderator for the collaborations, the assumptions, and experiments are guided with an eye toward technology application in the target environment.

Figure 7 shows a screen snapshot from the wiki page of an experiment conducted with a candidate solution (using ASCENT, an algorithm from Vanderbilt University) that was used to analyze a subset of the problem in Figure 6.
Enhancing Community Building

Foundational technology to support the discovery of suitable participants in the various SPRUCE communities is currently under development. This technology is based on the notion of affinity relationships between challenge problems, solutions technologies, and personnel interests and publications. The automated collection and construction of affinity relationships will allow SPRUCE to offer query mechanisms such as “since person A is interested in this problem, who else may be interested?”, or “given the challenge problems description, construct a list of leading researchers that have published in specified leading journals on related subjects.” SPRUCE users can then use the results of such queries to construct invitations to potential collaborators.

Other features under consideration include social networking and community communication mechanisms to facilitate a virtual community, such as instant messaging, presence information, text and media chat.

Getting Involved

If you wish to participate in SPRUCE, please visit the SPRUCE portal (www.sprucecommunity.org) and/or contact the authors. All forms of participation and contributions are welcome – be it through participating with existing communities of interest, establishing and leading new communities of interest, or providing representative DoD artifacts to be shared with the SPRUCE community.

Figure 7: Example screen from a candidate solution documenting the results of an experiment
Conclusion

Experimentation and collaboration around representative challenge problems in military and aeronautics domains have the potential to bridge the divide between various stakeholders in the DoD ecosystem—a challenge heretofore achieved only through ad hoc and serendipitous engagement between participants. SPRUCE offers an exciting opportunity to address this challenge head on, by providing a platform to promote the desired experimentation and collaboration. The SPRUCE program will undertake a foundational effort in getting an initial set of communities started. Members of the SPRUCE community have the power to sustain and shape the evolution of SPRUCE in the years ahead.

About the Authors

Mr. Patrick Lardieri is a Manager of Advanced Software Technology Research at Lockheed Martin Advanced Technology Laboratories. Mr. Lardieri has over ten years experience in engineering, implementing, designing, testing, demonstrating, and coordinating prototypes in a research and development environment.

Dr. Richard Buskens is a Manager of Advanced Software Technology Research at Lockheed Martin Advanced Technology Laboratories. He has over 20 years of advanced and applied research experience, primarily focused on software producibility activities, and a proven track record for carrying innovative research ideas from concept stage through to product-quality prototypes.

Mr. Srini Srinivasan is a Technology Consultant at Lockheed Martin Advanced Technology Laboratories. Srini has over 20 years of experience in developing and commercializing new technologies.

Mr. Steven Drager is the Advanced Computing Architecture Core Technical Competency lead as well as the technical advisor for the Computing Applications Technology Branch of the Air Force Research Laboratory Information Directorate. Mr. Drager has worked over 20 years at AFRL beginning in reliability physics where he worked wafer-level testing for oxide breakdown, hot carrier degradation, and electromigration.

Mr. William McKeever is a Computer Scientist at the Air Force Research Laboratory Information Directorate in the Computing Applications Technology Branch (AFRL/ RITB). He has worked for AFRL for over five years as a researcher and program manager.

Recent Technical Reports from DACS

  Released: 9/30/2007

- Enhancing the Development Life Cycle to Produce Secure Software
  Released: Version 2.0, October 2008
  https://www.thedacs.com/techs/enhanced_life_cycles/

  Released: August 2007

  Released: 9/30/2007

PDF versions of these reports can be downloaded free of charge by registered DACS visitors who are logged in.

Hard copies can be ordered (for fee) from the DACS store online at https://store.thedacs.com/
The STN is a theme-based quarterly journal. In the past DACS has typically solicited specific authors to participate in developing each theme, but we recognize that it is not possible for us to know about all the experts, programs, and work being done and we may be missing some important contributions. Therefore, beginning in 2009 DACS is adopting a policy of accepting articles submitted by the software professional community for consideration.

DACS will review articles and assist candidate authors in creating the final draft if the article is selected for publication. Note that DACS does not pay for articles published. Note also that submittal of an article constitutes a transfer of ownership from the author to DACS with DACS holding the copyright.

Although the STN is theme-based, we do not limit the content of the issue strictly to that theme. If you submit an article that DACS deems to be worthy of sharing with the community, DACS will find a way to get it published. However, we cannot guarantee publication within a fixed time frame in that situation. Consult the theme selection page and the Author Guidelines located on the STN web site (see https://www.softwaretechnews.com/) for further details.

To submit material (or ask questions) contact news-editor@dacs.dtic.mil

Tentative Themes for 2009 are:

- Earned Value
- Software Testing
- Project Management
- Model Driven Development
The first 50 people to send in a completed survey will receive a FREE DoD/IT Acronym CD from the DACS.

This valuable CD-ROM contains over 9,000 Department of Defense and Information Technology acronyms. There are hundreds of acronym lists available but none are as well done as this CD AND specifically targeted towards DoD and Information Technology. This unique-shaped CD-ROM plays in your computer’s regular, hub-mounted, CD drive. You’ll use this great resource over and over again. It’s FREE, just for filling out our brief survey on the next page!

Data & Analysis Center for Software (DACS)

http://iac.dtic.mil/dacs/
SOFTWARE TECH NEWS FEEDBACK

1. Which volume of the Software Tech News did you receive?  STN Vol. 12, No. 1

2. When did you receive the newsletter? (month/year)

3. How satisfied were you with the CONTENT of the newsletter? (Article Quality)
   - [ ] Very Satisfied
   - [ ] Satisfied
   - [ ] Neither Satisfied nor Dissatisfied
   - [ ] Dissatisfied
   - [ ] Very Dissatisfied

4. How satisfied were you with the APPEARANCE of the newsletter?
   - [ ] Very Satisfied
   - [ ] Satisfied
   - [ ] Neither Satisfied nor Dissatisfied
   - [ ] Dissatisfied
   - [ ] Very Dissatisfied

5. How satisfied were you with the OVERALL QUALITY of the newsletter?
   - [ ] Very Satisfied
   - [ ] Satisfied
   - [ ] Neither Satisfied nor Dissatisfied
   - [ ] Dissatisfied
   - [ ] Very Dissatisfied

6. How satisfied were you with the ACCURACY of the address on the newsletter?
   - [ ] Very Satisfied
   - [ ] Satisfied
   - [ ] Neither Satisfied nor Dissatisfied
   - [ ] Dissatisfied
   - [ ] Very Dissatisfied

7. Approximately how much of the newsletter do you read?
   - [ ] The entire issue
   - [ ] Most of the content
   - [ ] About half the content
   - [ ] Briefly Skimmed
   - [ ] Didn’t Read

8. Would you read this newsletter in an E-mail newsletter format?
   - [ ] Definitely
   - [ ] Probably
   - [ ] Not Sure
   - [ ] Probably Not
   - [ ] Definitely Not

9. How did you request the product or service?
   - [ ] Phone Call
   - [ ] E-mail
   - [ ] DACS Website
   - [ ] Subscription Form
   - [ ] Other __________________________

10. Would you recommend the Software Tech News to a colleague?
    - [ ] Definitely
    - [ ] Probably
    - [ ] Not Sure
    - [ ] Probably Not
    - [ ] Definitely Not

11. What topics would you like to see this newsletter devoted to?
    ___________________________________________________________________________________
    ___________________________________________________________________________________
    Comments (optional) ___________________________________________________________________
    ___________________________________________________________________________________

REGISTER FOR THE FIRST TIME | UPDATE CURRENT SUBSCRIPTION

Name: ________________________________  Position/Title: ________________________________
Organization: __________________________  Office Symbol: ______________________________
Address: ______________________________  City: ________________________________
State: __________________ Zip: __________  Country: ________________________________
Telephone: _____ - ______ - _______ Fax: _____ - ______ - _______ Email: _____________________
Functional Role: __________________________
Organization Type:  
   - [ ] Air Force
   - [ ] Army
   - [ ] Navy
   - [ ] Other DoD __________________________
   - [ ] Commercial
   - [ ] Non-Profit
   - [ ] Non-US
   - [ ] US Government
   - [ ] FFR&D
   - [ ] Other __________________________

*Note: You must give us your address to receive the CD.*
STN EDITORIAL BOARD

Ellen Walker
Managing Editor
ITT Corporation, DACS

Dan Ferens
Co-Editor
ITT Corporation, DACS

Philip King
Production Editor
ITT Corporation, DACS

Wendy Butcher
Graphic Designer
ITT Corporation, DACS

Paul Engelhart
DACS COTR
Air Force Research Lab

Morton A. Hirschberg
Editorial Board Chairman
Army Research Lab (retired)

Thomas McGibbon
DACS Director
ITT Corporation, DACS

Dr. Kenneth E. Nidiffer
Software Engineering Institute

Dennis Goldenson
Software Engineering Institute

John Scott
Mercury Federal Systems

Distribution Statement:
Unclassified and Unlimited

DACS
P.O. Box 1400
Rome, NY 13442-1400
Phone: 800-214-7921
Fax: 315-838-7130
E-mail: dacs@dtic.mil
URL: http://www.thedacs.com/

ABOUT THE SOFTWARE TECH NEWS

ADVERTISEMENTS

The Software Tech News is now accepting advertisements for future newsletters. In addition to being seen by the thousands of people who subscribe to a paper copy, an electronic version is available at the DACS website, exposing your product, organization, or service to hundreds of thousands of additional eyes every month.

For rates and layout information contact: news-editor@dacs.dtic.mil

COVER DESIGN

Wendy Butcher
Graphic Designer
ITT Corporation, DACS

Source for chart graphic on cover:
John Rusk

ARTICLE REPRODUCTION

Images and information presented in these articles may be reproduced as long as the following message is noted:

“This article was originally published in the Software Tech News, Vol.12, No.1 April 2009. Requests for copies of the referenced newsletter may be submitted to the following address:

Philip King, Production Editor
Data & Analysis Center for Software
P.O. Box 1400
Rome, NY 13442-1400

Phone: 800-214-7921
Fax: 315-838-7130
E-mail: news-editor@dacs.dtic.mil

In addition to this print message, we ask that you notify DACS regarding any document that references any article appearing in the Software Tech News.

ABOUT THIS PUBLICATION

The Software Tech News is published quarterly by the Data & Analysis Center for Software (DACS). The DACS is a DoD sponsored Information Analysis Center (IAC), administratively managed by the Defense Technical Information Center (DTIC). The DACS is technically managed by Air Force Research Laboratory, Rome, NY and operated by ITT, Advanced Engineering and Sciences Division.
<table>
<thead>
<tr>
<th>Title</th>
<th>Authors</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tech Views</td>
<td>by Dan Ferens and Ellen Walker, Co-Editors, DACS Software Tech News</td>
<td>3-6</td>
</tr>
<tr>
<td>EVM in Government: Striking a Balance</td>
<td>by Wayne F. Abba, Abba Consulting</td>
<td>8-11</td>
</tr>
<tr>
<td>Applying Earned Value Management to Software Intensive Programs</td>
<td>by Robert P. Hunt and Dan Galorath (Galorath Incorporated), and Paul J. Solomon (Performance-Based Earned Value)</td>
<td>12-18</td>
</tr>
<tr>
<td>Earned Value for Agile Development</td>
<td>by John Rusk, Optimation Ltd</td>
<td>20-27</td>
</tr>
<tr>
<td>Defense Acquisition University’s Integrated Program Management Model</td>
<td>by David C. Bachman, Defense Acquisition University</td>
<td>29-35</td>
</tr>
<tr>
<td>SPRUCE: Systems and Software Productibility Collaboration and Experimentation Environment</td>
<td>by Patrick Lardieri, Rick Buskens, Srini Srinivasan, Lockheed Martin Advanced Technology Laboratories, William McKeever and Steven Drager, Air Force Research Laboratory</td>
<td>36-43</td>
</tr>
</tbody>
</table>