Manage your projects in guaranteed compliance with the CMMI — Now!

Don’t waste valuable time and resources developing CMMI-compliant processes from scratch when there is a proven approach that guarantees success. With **processMax**®, you begin operating in compliance immediately: no process development is required!

**processMax** is a complete project management system, integrated with Microsoft Project, and is guaranteed by **pragma** SYSTEMS to be compliant with CMMI-DEV.

With **processMax**, managers and their teams efficiently collaborate with step-by-step procedures, integrated document management, risk management, automated workflow, and automated measurement and reporting. **processMax** increases productivity, reduces defects, and manages risk.

Now available as a hosted service for both our subscription and perpetual licenses, **processMax** is more affordable than ever. We manage the server, installation, updates, and upgrades.

More than 70 organizations have passed Level 2 and Level 3 appraisals with **processMax**, at a fraction of the time and expense required by traditional methods.

Please contact us to learn how **processMax**, can help you achieve your compliance goals.

**pragma** SYSTEMS CORPORATION

www.pragmasystems.com
703-796-0010
info@pragmasystems.com

How long can you wait for CMMI® Compliance?
We all understand the importance of data to manage any program, whether it is estimated data for planning purposes or actual data for monitoring project progress and performance. The importance of data in managing systems acquisition programs, from a software intensive systems perspective, is particularly crucial when:

- Acquisition program requirements are complex and diverse
- Large and complex software and systems development teams are being used, including teams of teams
- Users specify requirements and accept systems, but are not part of the development process
- Complex and difficult to understand system architectures are used
- Many complex legal issues exist (e.g., contracts, copyrights, warranties)

Cost data, in particular, plays a key role in acquisition planning, making affordability trade-off decisions, and monitoring acquisition performance. Cost estimation is a key activity underlying all of systems acquisition management, where in many cases estimates need to be made long before critical decisions have been finalized or much is known about all of the key requirements. Historical cost data, or models built on historical data, are used to generate cost estimates. However, interpreting what is in the data, and the relationships of data to outcomes, remains a key challenge.

This issue of the DACS Journal looks at the role of cost estimation and data in systems acquisition management. We have assembled a group of leading subject matter experts in systems acquisition and cost estimation to share their knowledge.

Our lead article, *The Need for “Acquisition Visibility”*, written by Mr. Mark Krzysko, the Deputy Director of the Enterprise Information and Office of the Secretary of Defense Studies, provides an outstanding discussion and sets the stage for describing the need for data – data as a service - to support the “Better Buying Power” initiative of the DoD, and how that need is being addressed.

The DACS has, for many years, endorsed and implemented this vision of data as a service. With this issue of the DACS Journal, in the article *The DACS Software & Systems Cost and Performance Analysis Toolkit*, we are announcing the addition of a new data service and resource to provide representative cost data to support the software intensive systems acquisition and cost estimation communities, in line with the vision of access to key data within the Better Buying Power initiative. We are partnering with the University of Southern California (USC) Center for Systems and Software Engineering to develop this toolkit, leveraging their knowledge in development of the COCOMO-based models.

Several parametric cost estimation experts provide their insight on effective cost estimation methods using historical data throughout the systems acquisition lifecycle:

- Throughout the systems acquisition lifecycle, the Government has to develop independent systems cost estimates. William Roetzheim, in his article *Parametric Modeling to Support Systems Acquisition*, describes the use and benefits of parametric models for estimation, starting from the very earliest phases of the lifecycle.
- Arlene Minkiewicz of PRICE Systems, and a frequent author for the DACS Journal, describes a strategy for being able to defend cost estimates in her article *Selling Your Cost Estimate*. Having good data is an important factor.
- Another article describing the importance of historical data and its use for estimation and understanding of the impacts of changes is the article by Kate Armel, from Quality Software Management, Inc., entitled *History is the Key to Estimation Success*. In this article she shows, from a database of over 1,000 historical projects, the impact on cost and quality of small changes to project team size.

We also include an article addressing current research in cost estimation. A serious challenge to estimators is developing credible estimates well before a system solution is conceived. Addressing this, the SEI’s Cost Estimation Research Group describes their work in *An Innovative Approach to Quantifying Uncertainty in Early Lifecycle Cost Estimation*. In it, they describe their proposed methods for pre-Milestone A cost estimation.

In the article *Outcome-Based Software Acquisition*, Roger Stewart and Lew Priven describe the use of standards-based Inspections and supporting tools to achieve an auditable and actionable outcome-based acquisition process.

We close out this issue of the DACS Journal with the article *A Roadmap for Reforming the DoD’s Acquisition of Information Technology* from John M. Gilligan, former Air Force Chief Information Officer (CIO), who describes an effective five-point program to improve DoD IT acquisition processes.
The Need for “Acquisition Visibility”
By Mark Krzysko and Brian Baney

On September 14, 2010, then Under Secretary of Defense for Acquisition, Technology and Logistics (USD(AT&L)), Dr. Ashton Carter, provided guidance to acquisition professionals regarding the “Better Buying Power” initiative. The Department of Defense (DoD) introduced Better Buying Power – with broad support from senior DoD leadership – to improve acquisition value through increased competition, tradecraft, productivity and incentives. Better Buying Power also focused heavily on how the Department managed its complex acquisition portfolio by challenging the workforce to improve critical acquisition management, oversight and processes. The Department would achieve better acquisition management and processes, according to Better Buying Power, with a greater focus on cost, affordability and performance – elements that were becoming increasingly important as DoD faced growing scrutiny on its acquisition programs.

The timing of Better Buying Power was no surprise. In 2010 the Department’s Major Defense Acquisition Program (MDAP1) portfolio grew to over $1.7 trillion across the life of the programs, an almost 60% increase from acquisition numbers reported ten years earlier.2 The $1.7 trillion did not account for certain operations and sustainment costs, the Military Department’s smaller acquisition programs, or Major Automated Information Systems, adding billions to an already incredible portfolio. DoD’s acquisition processes were also the subject of ongoing Congressional scrutiny and numerous Government Accountability Office (GAO) and DoD Inspector General (IG) investigations.3 Highlighting significant management and oversight issues, the GAO claimed that development costs for acquisition programs were as much as 42% higher than original estimates, with program delays averaging 22 months.4 While DoD worked closely with GAO to review their conclusions, the scope of the portfolio is real.

Better Buying Power provided a strategic foundation to implement change, but acquisition leadership still needed to answer a tough but appropriate question: did DoD have the right information or systematic processes to address acquisition management, oversight and process complexities and achieve efficiencies that provided better value to the Warfighters and taxpayers? In the September 14, 2010, Better Buying Power guidance memorandum to acquisition professionals, Dr. Carter emphasized the use of data to answer this question.

“We have analyzed data on the Department’s practices, expenditures, and outcomes and examined various options for changing our practices. We have sought to base the specific actions I am directing today on the best data the Department has available to it. In some cases, however, this data is very limited. In these cases, the guidance makes provision for future adjustments as experience and data accumulate so that unintended consequences can be detected and mitigated.”5

The message was clear: data and information were key to managing, overseeing and streamlining processes within the acquisition portfolio, but DoD would require diligence to obtain it. Data offered innovative perspectives on acquisition processes, delivering the necessary insight into acquisition cost, performance, affordability and other critical elements. Empowered with data, DoD leadership could report, analyze, and make informed decisions on the Department’s complex acquisition portfolio. This is the focus of this article.

Prior to Better Buying Power, the Director of Acquisition, Resources and Analysis (ARA), an organization under the

---

1 10 U.S.C. 2430 defines a Major Defense Acquisition Program as a program that the Secretary of Defense defines as a major program (non classified), estimates that total expenditures for research, development, test and evaluation will exceed $365,000,000, or estimates that total expenditures for procurement will exceed $2,190,000,000.

2 Budget data from the Defense Acquisition Management Information Retrieval System

3 A listing of GAO and IG reports on defense acquisition issues and investigations can be found at http://www.gao.gov and http://www.dodig.mil.


USD(AT&L), led a parallel effort to support the reformation of acquisition management, oversight and departmental business processes. The Director, ARA directed a small team to focus on using structured data to provide insight on how the Department viewed and managed its acquisition portfolio. Key areas of focus were data governance and using data as a service. In conjunction with the Military Departments, the team demonstrated it could identify authoritative sources of major acquisition information; have consistent, semantic definitions across the Department; measure data for accuracy, reliability and availability; and provide it to acquisition leadership for use in any visual tool giving them data-driven insight into the major acquisition portfolio.

These efforts were part of a pilot that went from concept to execution in under 60 days – a short order for a complex Department of Defense. The team extracted performance data elements of 12 major weapons programs and presented this data to leadership within a service-oriented environment. DoD leadership would manage this service Department-wide with the goal of supporting reporting, analysis and decision making. Although the pilot represented a small subset of acquisition data and programs, it demonstrated two important achievements: first, DoD could manage and govern acquisition information with technologies and techniques such as Service Oriented Architecture (SOA) and semantics; second, DoD could provide data and information as a service in a near real-time environment. The ability for this team to reach across DoD and work closely with the Military Departments showed transcendence of organizational boundaries, infrastructures, organizational prerogatives, and security models. Acquisition data – once held tightly by those processes – was now transparent and understood. The USD(AT&L) saw the potential structured data could have on acquisition management, oversight and business processes and expanded the pilot to determine its full capability. Additionally, he sought an on-demand environment that could provide data across the enterprise seamlessly and efficiently. This capability became known as Acquisition Visibility (AV), which transitioned from pilot to implementation, formally effective in July 2009. The number of acquisition programs covered under Acquisition Visibility tripled during this transition and the team worked across the Department to apply structured acquisition data to other uses. They began to automate reporting to immediately comply with internal and Congressional reporting requirements and reused data to develop multiple reports at the same time. In support of other organizations, including the Office of Performance Assessments and Root Cause Analyses, DoD began to explore the use of data analytics, using business intelligence tools to provide dramatically better insight to leadership about how DoD oversaw and managed its acquisition programs. This also provided leadership awareness into how DoD’s industry partners were performing against their requirements and obligations. As data requirements increased, Acquisition Visibility quickly became the single source for acquisition information across the Department.

An Effective Capability Through Agility

While Acquisition Visibility provides important information to acquisition decision-makers, it is more than a technical capability which shares information across the Department. The service is the authoritative source for acquisition information, acquisition information definition and the foundation for how acquisition leadership oversees, manages, reports and analyzes its major acquisition portfolio. It delivers the core service for the USD(AT&L)’s acquisition oversight, accountability and strategic decision making. Acquisition Visibility leverages internal net-centric services from OSD defense agencies as well as engineering and technical expertise from the Military Departments and industry partners. Components now provide accurate, structured, acquisition information – via AV SOA as the core element of net-centricity – immediately to the USD(AT&L) and senior decision makers.

Acquisition Visibility does not impose uniform business processes across the DoD acquisition community, replace current applications and tools or implement an “IT solution”. It benefits from best practices, SOA and existing technical infrastructure to gain access to transparent, acquisition data without having to maintain a single data strategy or common infrastructure – each Component optimizes their business processes and leverages technologies as appropriate. Data governance ensures access to Component-managed acquisition information and defines data for consistency, accuracy and reliability across the Department. Governance also establishes definitive semantic meaning to the business rules so Components share acquisition information consistently within the AV environment. This enables Components, leadership and acquisition stakeholders to view and measure data uniformly and promotes information quality assurance when it comes to data reporting and analysis.

Multiple delivery teams support Acquisition Visibility providing functional, technical and data expertise. Acquisition Visibility now covers all major defense acquisition programs, by sharing sustainment, operating costs and earned value management data among other acquisition information. AV leadership has transitioned the service into a production
environment and grew the number of acquisition data elements to over 700 bringing additional, more relevant information to acquisition decision makers. The Department will add the Major Automated Information Systems and non-MDAP acquisition programs to this information paradigm in the near future. Figure 1 below provides a visual of Acquisition Visibility and its core services. In support of Acquisition decision making, DAES6, SAR7, UCR8, DAB9 and APB10 are legislative reporting requirements or decisions forums that leverage acquisition information via Acquisition Visibility.

A Case Study for Change

Acquisition Visibility is a case study in cross-organizational leadership, the use of best business practices, and technical competency within the Federal Government. Implementing this service required a significant amount of leadership, change management, and alignment of disparate defense organizations from Acquisition Visibility delivery teams. Teams constantly engaged and collaborated with acquisition organizations across the Department. They remained flexible to address leadership’s ever-changing data, service and technical requirements. Support constantly evolved as Acquisition Visibility adapted to and embraced cutting-edge technologies and concepts improving performance. In addition, AV leadership deliberately used agile methodologies to create synergies among delivery teams, including the Components, to meet strategic, functional, data and technical requirements more effectively.

Figure 1: Strategic View of the Acquisition Visibility service

Acquisition leadership agrees this remains an optimal approach despite the need for perpetual agility. By having instant access to authoritative acquisition information, leadership has insight into major acquisition programs more effectively. Performance data ensures Components are meeting cost, budget and specification targets while offering clues as to why certain programs may not be meeting their performance objectives. Data-driven decisions will help the Department address and overcome current and future acquisition challenges, whether organizational, budgetary or technical. The Department can reliably report on the performance and status of each major acquisition program to stakeholders – and Congress – to improve transparency. The ability to capture growing amounts of acquisition information
exposes the Department to the strengths of data analytics, providing leadership with even more powerful information to drive critical decision making and oversight in the future.

to be agile with a focus on immediately – both effectively and efficiently – responding to acquisition leadership.

### Table 1

<table>
<thead>
<tr>
<th>Action or Activity</th>
<th>Impact on Acquisition Management and the Acquisition Portfolio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development of an information sharing framework</td>
<td>• Structured acquisition information to manage the $1.7 trillion acquisition portfolio&lt;br&gt;• Data serves as the foundation for AT&amp;L reporting and analysis&lt;br&gt;• Measures for implementing the (USD)AT&amp;L’s Better Buying Power Initiative&lt;br&gt;• Compliance with statutory reporting, organizational and analytical requirements&lt;br&gt;• A service that shares acquisition information to leadership within an on-demand environment versus ad hoc briefings</td>
</tr>
<tr>
<td>Deliberate and perpetual use of agile methodologies</td>
<td>• Requirements definition to delivery within 60 days&lt;br&gt;• Collaboration among all delivery teams produces a shared services mentality&lt;br&gt;• Ability to implement new information requirements consistently and quickly across all delivery teams and the Department&lt;br&gt;• Mitigate single points of failure through the inclusion of delivery teams in all efforts&lt;br&gt;• Achieve the ultimate goal of being responsive to the customer (USD(AT&amp;L))</td>
</tr>
</tbody>
</table>

Viewing Acquisition Visibility from strategic, functional and technical perspectives provides insight as to why the Department created this service to be the authoritative source for acquisition information and the primary tool to improve management, oversight and decision making on the acquisition portfolio. These perspectives highlight how and why specific techniques were used, their impact to operations, and the benefits of using these strategies, so other organizations can replicate similar techniques across Department.

**Strategic Perspective:** Acquisition Visibility was an effective means to provide structured, and eventually unstructured, acquisition data and reports to leadership and acquisition stakeholders. Use of acquisition information – as a service – exploded as DoD leadership realized the value of authoritative, reliable acquisition information to manage, oversee and support informed decision making on major acquisition expenditures. Consequently, Acquisition Visibility’s vision and mission has adapted since its pilot/implementation phase to account for the USD(AT&L)’s data requirements and growing uses for acquisition information. DoD leadership expects and requires timely, “ground truth” information regarding the acquisition portfolio, and Acquisition Visibility evolved into the service that provides it to them. What was once organizationally consuming to manage (data), is now a real-time data service executed by strong governance and technical competency. To implement this type of service, AV leadership needed a strategic information framework to manage acquisition information across the Department (Table 1). This framework needed to identify data requirements, support both functional and technical processes, and posses the appropriate infrastructure to connect with acquisition leadership. This framework needed to be agile with a focus on immediately – both effectively and efficiently – responding to acquisition leadership.

**Functional Perspective:** Customers who use service providers rarely consider the processes or technical infrastructure needed to meet their requirements; they simply demand the service they request as expeditiously as possible. A key component to Acquisition Visibility’s success is the delivery teams’ focus on end-users’ (customers) requirements for acquisition information. Thus, the service provides the right information through functions that are easy to use while meeting leadership’s demands at the same time. USD(AT&L) leadership ensures collaboration across organizational boundaries to obtain acquisition information. This collaborative environment lets delivery teams not only understand the type of information end-users need, it also allows them to understand how end-users would like to obtain, access and see the data. Governance and enterprise-level business process modeling (Table 2) streamlines and aligns business processes and rules across the Department to allow Acquisition Visibility to meet acquisition data requirements more effectively (Components maintain the flexibility to management their own internal data processes as appropriate).

**Technical Perspective:** AV leadership’s insistence to be “world class” in technical delivery is the nucleus for sharing acquisition information (Table 3). Acquisition Visibility is an information service that relies heavily on systems and software engineering and other technical best practices such as cloud computing and open source software to achieve its objectives. The use of technical best practices and services provides delivery teams flexibility when managing technical requirements, reduces Acquisition Visibility’s footprint and organizational impacts across DoD, and provides delivery teams access to cutting-edge tools to address unique, changing
information demands. This information service is a federation of software and data of various servers across OSD and the Military Departments to promote internal cooperation among Components. Strong information stewardship ensures the best value regardless of where the servers are located. The use of both organic and private industry expertise with systems and software engineering, SOA and data governance enables Acquisition Visibility to provide leadership strong support despite the varying complexities of security, architecture and infrastructure. As the amount of information shared across the service continues to expand, DoD has looked to more automated and accurate ways to share acquisition data. Just last year, acquisition leadership worked with Components to implement a Department-wide “publish and subscribe” process. This process allows leadership to pull data from the various acquisition portfolios making data more readily available to leadership and promoting better data quality among the Components. As these types of techniques mature, Acquisition Visibility can continue to serve as a technical model that utilizes the most advanced technologies and best practices while reducing costs and the overall technical footprint.

Figure 2 below depicts Acquisition Visibility’s maturity and growth since its inception as a pilot program three years ago. Three perspectives measure Acquisition Visibility’s maturity. First is the functional growth and improvements to Acquisition Visibility (Y-axis). Second is the increase of acquisition information shared across DoD as acquisition leadership took a combination of organizational and technical steps to improve data delivery (X-axis). Third is a combination of both perspectives: DoD realizes capability synergies and achieves technical transformation.

<table>
<thead>
<tr>
<th>Action or Activity</th>
<th>Impact on Acquisition Management and the Acquisition Portfolio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use of SOA as the core of net-centricity</td>
<td>• Creates a comprehensive, shared services environment with the ability to get acquisition information to leadership instantaneously</td>
</tr>
<tr>
<td>Focus on leveraging services and tools that</td>
<td>• Access to existing DoD infrastructure via enterprise messaging reducing hardware costs to OSD and Military Departments</td>
</tr>
<tr>
<td>already exist</td>
<td>• Lightweight and flexible information sharing that complies with Joint specifications</td>
</tr>
<tr>
<td>Publish and subscribe approach</td>
<td>• Unifies the Services and Component acquisition community with a federated information sharing approach that reduce costs and significantly improves timeliness and reliability</td>
</tr>
</tbody>
</table>

Table 2

<table>
<thead>
<tr>
<th>Action or Activity</th>
<th>Impact on Acquisition Management and the Acquisition Portfolio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enterprise-level data governance</td>
<td>• Data transparency and accountability for informed decision-making on acquisition programs</td>
</tr>
<tr>
<td></td>
<td>• The ability to access and obtain Component-managed data and infrastructure specifications</td>
</tr>
<tr>
<td></td>
<td>• Consistent and defined acquisition data for multiple uses across the Department</td>
</tr>
<tr>
<td></td>
<td>• A structure for working within a disparate and bureaucratic environment</td>
</tr>
<tr>
<td></td>
<td>• Identifies the authoritative sources of acquisition data and where they maintain the data</td>
</tr>
<tr>
<td>Adoption of business process modeling</td>
<td>• Consistent processes for working with acquisition portfolios across DoD</td>
</tr>
<tr>
<td>techniques</td>
<td>• Easily incorporates new stakeholders or acquisition information into the AV environment</td>
</tr>
<tr>
<td></td>
<td>• Scalable processes that account for the complexities of meeting unique data requirements</td>
</tr>
<tr>
<td></td>
<td>• Process clarity and cross-team integration by illustrating roles and activities across DoD</td>
</tr>
</tbody>
</table>

Table 3
Overcoming Challenges for an Information Acquisition Future

No technology, organization or process is a perfect solution and DoD’s work to implement Acquisition Visibility came with numerous challenges. While AV leadership deliberately used the strategic, functional and technical activities to mitigate anticipated issues, delivery teams continue to work the many common problems today. Data transparency has always been a primary obstacle to achieving Acquisition Visibility. Organizational tendencies to compartmentalize acquisition data – reflecting their authority to acquire and manage their own major programs – and a hesitancy to share critical acquisition information with outside organizations are being overcome. However, the clarity that Acquisition Visibility brings to identify authoritative sources of data, determine end-user requirements, and meet multiple data and information demands offers a bright future for DoD’s acquisition portfolio.

About the Author

Mark Krzysko serves as the Deputy Director, Enterprise Information and Office of the Secretary of Defense Studies. In this senior leadership position, he directs data governance, technical transformation, and shared services efforts to make timely, authoritative acquisition information available to support oversight of the Department of Defense’s major programs, a portfolio totaling more than $1.7 trillion of investment funds over the lifecycle of the programs. Mr. Krzysko oversees Federally Funded Research and Development Centers and University Affiliated Research Centers.

Brian Baney is a Managing Consultant in IBM Global Business Services’ U.S. Federal Division. He specializes in supply chain management, national security and public policy working within the homeland security and defense industries.
Two Great Reliability Solutions from the RIAC & DACS.

System Reliability Toolkit

The RIAC/DACS System Reliability Toolkit provides technical guidance in all aspects of system reliability, allowing the user to understand and implement techniques to ensure that system and product designs exhibit satisfactory hardware, software and human reliability, and to minimize the inherent risks associated with deficiencies in system reliability.

To purchase, please contact:
The Reliability Information Analysis Center
100 Seymour Road
Suite C-101
Utica, NY 13502
1-877-363-RIAC
http://theRIAC.org

The DACS Software Reliability Sourcebook

This first edition of the DACS Software Reliability Sourcebook provides a concise resource for information about the practical application of software reliability technology and techniques. The Sourcebook is divided into nine major sections, plus seven supporting appendices.

To purchase, please contact:
The Data & Analysis Center for Software
100 Seymour Road
Suite C-102
Utica, NY 13502
1-800-214-7921
http://thedacs.com
Satisfy Your Quality Obsession

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. Streamline communication, improve traceability, achieve compliance, and deliver quality products with Seapine Software’s scalable, feature-rich application lifecycle management solutions:

- **TestTrack Pro** — Development workflow and issue management
- **TestTrack TCM** — Test case planning and tracking
- **Surround SCM** — Software configuration management
- **QA Wizard Pro** — Automated functional and regression testing

Designed for the most demanding software development and quality assurance environments, Seapine’s flexible cross-platform solutions adapt to the way your team works, delivering maximum productivity and saving you significant time and money.

Visit [www.seapine.com/gsa](http://www.seapine.com/gsa)
The goal of Data and Analysis Center for Software (DACS) Software & Systems Cost and Performance Analysis Toolkit (S²CPAT) is to capture and analyze software and software engineering data from completed software projects that can be used to improve a) the quality of software-intensive systems and b) the ability to predict the development of software-intensive systems with respect to effort and schedule.

The toolkit allows users to search for similar software projects and use the data to support:

1. Rough order of magnitude estimates\(^1\) for software development effort and schedule
2. Project planning and management: life cycle model information, key risks, lessons learned, templates, estimation heuristics
3. Software engineering research.

Overview of Current Toolkit Capabilities

Currently, the Software & Systems Cost and Performance Analysis Toolkit can display statistics about all of the software projects in the database as well as allow users to specify search criteria to select software projects of interest based on project size and application domain. The current version of the toolkit provides some standard statistical analyses of the selected projects.

In addition, there is a data submission capability that allows users to submit additional projects to the toolkit so that the toolkit can grow and better reflect changing software development approaches, languages, and tools.

Overview of Statistical Analysis Capabilities

Before searching for information about specific projects in the toolkit, the user can first review general information about the toolkit contents related to distribution of projects.

---

\(^1\) Toolkit will provide links to participating cost model vendors for more detailed/precise estimates and additional support.
DACS SOFTWARE & SYSTEMS COST AND PERFORMANCE ANALYSIS TOOLKIT (CONT.)

by software size and effort, application domain, software operating environment, and development languages, as shown in Figure 1. The upper left chart in Figure 1 shows an example scatter chart with software size on the y-axis and effort (labor hours) on the x-axis. The upper right chart in Figure 1 shows a distribution of the toolkit projects across application domains (note that the current domains primarily reflect DoD types of projects). The two lower charts illustrated project distributions by operating environment (primarily military environments/platforms) and development language.

Searching Function with Output Statistics

To select a subset of the projects in the toolkit for analysis, users can use the query parameters for software size (specific value with an optional ± range, application domain, primary development language, and operating environment. User queries can specify one or more of these query parameters using the fields shown in Figure 2. The toolkit returns an aggregated response to the user query based on the toolkit records that match the specified parameters. Note that in order to protect the identity of the specific projects in the toolkit, results are provided only for those queries that result in at least four records with no one organization providing more than 50% of the data points.

After the submission, the system will pop up messages as shown in Figure 5. The user can either choose to update the previously submitted data or submit another new data point.

When a user submits a set of data for incorporation into the toolkit, the validation team analyzes the data to ensure that the proper counting rules were used and that the data can be aggregated with the rest of the toolkit data to support future toolkit queries. Once the user-submitted data is validated, it is incorporated into the toolkit. Currently, the DACS toolkit team is working on a reward system that will provide custom analyses and reports for those organizations that contribute data directly to the toolkit.

The following Government personnel are to be acknowledged for their efforts in sanitizing and releasing the Software Resources Data Report Data:

- Wilson Rosa, Ph.D., Technical Advisor, Information Technology Division, Air Force Cost Analysis Agency
- Christopher Zember, Deputy Director, DoD Information Analysis Centers, Defense Technical Information Center
- Paul Engelhart, DACS Contracting Officers Representative, Air Force Research Laboratory Information Directorate

Figure 2. Searching Function

After the user inputs the query conditions, and clicks on “Search”, the statistics results (Average, Median, Min, and Max) and more direct visualization such as bar charts or pie charts for all the measures (Productivity, Size, Effort, Duration, Peak Staff, SW Effort Distributions, Staff Experience Distribution, CMMI Level, Operating Environments, Developing Process, Primary Language) are displayed for those projects for which the data is available. Figure 3 illustrates the format of this output using random test data.

Submitting and Updating Software Data

The data submission user input screens, illustrated in Figure 4, are primarily based upon information available from the DoD Software Resources Data Report (SRDR) forms [1] and COCOMOII parameters [2]. The explanation of each data item is displayed when the user hovers the mouse on associated with the field.

Initial Toolkit Contents

The initial release of the DACS Software & Systems Cost and Performance Analysis Toolkit will contain Software Resources Data Report (SRDR) data provided by the Air Force [1]. This data has been sanitized for public release by the US Department of Defense (DoD) and validated by a DoD-funded academic research team. (see sidebar) Efforts are also underway at the University of Southern California (USC) Center for Systems and Software Engineering (CSSE) to obtain permission from their affiliates to migrate the academic COCOMO II calibration data into the DACS Software & Systems Cost and Performance Analysis Toolkit. Efforts are also underway to include additional DoD SRDR data in the toolkit as it is approved for public release. Individual project data submitted to the toolkit will capture both the SRDR and COCOMO II fields so that a richer set of software development performance analyses can be supported in the future.
Planned Schedule

Currently the toolkit has been presented at several software measurement-related conferences and workshops and has evolved to a beta version as a result of these reviews and comments. Current plans are to:

- Make the beta version available for general review and comment by January 15, 2012
- Solicit comments/suggestions during the rest of January and early February 2012
- Incorporate comments late in February 2012

Figure 3. Searching Output Statistics
• Roll out the initial release for general access, populated with the SRDR data currently approved for public release in early March 2012.
To obtain additional information about accessing the beta or initial release version or to get the latest information on the toolkit, go to http://www.thedacs.com/databases/SSCPAT/.

References

About the Authors
Qi Li is a Ph.D candidate and research assistant at the Center for Systems and Software Engineering, University of Southern California, and his advisor is Dr. Barry Boehm. He is also the lead developer for S2CPAT. His current research interests mainly focus on Empirical Software Engineering, with special interests on Software Cost Estimation & Modeling, and Value-Based Software Engineering. His dissertation title is “Value-based, Dependency-aware Inspection and Testing Prioritization”. He has published over 10 papers in the software engineering field and was awarded the Best Paper Awards from HICSS 2010 and PROMISE 2011. He also interned with IBM Rational in 2010 and Galorath Incorporated in 2011. He obtained his master degree in 2008 from Institute of Software Chinese Academy of Sciences and bachelor degree in 2004 from Beijing Normal University, all in Computer Science.

Jo Ann Lane is a research assistant professor at the University of Southern California Center for Systems and Software Engineering, conducting research in the areas of software engineering, systems engineering, and system of systems engineering (SoSE). She was a co-author of the 2008 Department of Defense Systems Engineering Guide for Systems of Systems. Current areas of research include SoSE processes, SoSE cost modeling, and SoS constituent system interoperability. Prior to her current work in academia, she was a key technical member of Science Applications International Corporation’s Software and Systems Integration Group for over 20 years, responsible for the development and integration of software-intensive systems and systems of systems. She received her PhD in systems engineering from the University of Southern California and her Master’s in computer science from San Diego State University.
History is the Key to Estimation Success
By Kate Armel, QSM

There are known unknowns…
But there are also unknown unknowns,
The ones we don’t know
We don’t know.

—Secretary of Defense Donald Rumsfeld, February 2002

It was late afternoon in April of 1999 when the phone in my office rang. The conversation went something like this:

“This software estimate just landed on my desk and I need to finish it by close of business today to support a fixed price bid.”

“What can you tell me about this project?”

“We’re rewriting an existing mainframe billing system developed in COBOL. The new system will be written in C++, so it should be much smaller than the old system.”

“Great–perhaps we can use the existing system as a rough baseline. How big is it?”

“I don’t have that information.”

“Will this be a straight rewrite, or will you add new features?”

“Not sure–the requirements are still being fleshed out.”

“What about resources? How many people do you have on hand?”

“Not sure–the team size will depend on how much work must be done… which we don’t know yet.”

“Can we use some completed projects to assess your development capability?”

“Sorry, we don’t have any history.”

“That’s OK–even without detailed information on scope, resources, or productivity we should still be able to produce a rough order of magnitude estimate based on relevant industry data.”

“Rough order of magnitude??! My boss will never accept that much risk on a fixed price bid! Isn’t there some general rule of thumb we can apply?”

Welcome to the world of software cost estimation where the things we know – the known knowns - are often outweighed by the things we don’t know. Numerous estimation methods exist. Scope is described using effort, delivered code volume, features, or function points. Expert judgment, Wideband Delphi, top down, bottom up, parametric and algorithmic models each have their determined champions. But regardless of method, all estimates are vulnerable to risk arising from uncertain inputs, requirements changes, and scope creep. Skilled estimators and better methods can reduce this risk, but they can’t eliminate it. Thus, the ability to identify and account for uncertainty remains a vital component of successful risk management.

Estimation Accuracy vs. Estimation Usefulness

How accurate is the average software cost estimate? Industry statistics vary as widely as the estimates they seek to measure. One oft-cited study – the Standish Group’s Chaos Report – concludes that only one third of software projects deliver the promised functionality on time and within budget. A later IEEE study noted several gaps in the Standish Group’s criteria for estimation accuracy:

…the [Standish] definitions don’t cover all possibilities. For instance, a project that’s within budget and time but that has less functionality doesn’t fit any category. … The Standish Group’s measures … neglect under runs for cost and time and over runs for the amount of functionality.

When studies rely on different definitions of estimation success or failure, we should expect their assessments of estimation accuracy to exhibit considerable variability. The existence of different standards raises an intriguing question: what makes an estimate “accurate”?
Most quality control measures for estimates compare estimated cost/effort, schedule, or scope to their actual (final) values. The problem with this formulation is that “accurate” estimates are an integral part of feasibility decisions made very early in the project lifecycle; long before anything but the most generic information about the system’s intended features or use can be known with reasonable certainty. The technologies used to implement the requirements may be unknown and the schedule, team size, required skill mix, and project plan have yet to be determined. As design and coding progress, the list of unknowns grows shorter and decreasing uncertainty about the estimation inputs lowers the risk surrounding the estimated cost, schedule, and scope. Unfortunately, most organizations must make binding commitments before detailed and reliable information about the project is available.

Given the degree of uncertainty surrounding early estimates – and the correspondingly broad range of possible time/effort/scope combinations - estimation accuracy may be less important than estimation usefulness. In an article for the ACM, Philip Armour explores the difference between these two concepts:

The commitment is the point along the estimate probability distribution curve where we promise the customer and assign resources. This is what we need to hit, at least most of the time. It is not a technical estimation activity at all but is a risk/return based business activity. It is founded on the information obtained from the estimate, but is not the estimate. Using Figure 3 as an example, if we needed an accurate commitment in the earliest (Initial Concept) phase based on how the diagram shows the project actually worked out, we would have had to commit at around a 75% probability. From the figure, committing to the “expected” result at Initial Concept would have led to a significant overrun beyond that commitment, and the project would have “failed.” We can consider the 50% (expected) result to represent the cost of the project and the 25% increment to the higher commitment level to represent the cost of the risk of the project.

Measures of estimation accuracy that treat an estimate as “wrong” or a project as “failed” whenever the final scope, schedule, or cost differ from their estimated values penalize estimators for something outside their control: the uncertainty that comes from incomplete information. We should measure deviations between estimated and actual project outcomes because this information helps us quantify estimation uncertainty and account for it in future estimates. But if measurement becomes a stick used to punish estimators, they will have little incentive to collect and use metrics to improve future estimates.

**Understanding and Assessing Tradeoffs**

An old project management maxim succinctly summarizes the choices facing software development organizations: “You can have it fast, cheap, or good. Pick two.” Given that estimates (and therefore, commitments) are made early in the project lifecycle when uncertainty is high and the range of possible solutions is still wide, how do we select plans with a high probability of success? A thorough understanding of management tradeoffs can help. The idea behind the infamous Project Management Triangle is simple but powerful: the tradeoffs between software schedule, effort or cost, and quality are both real and unforgiving. Thanks to the work of pioneers like Fred
Brooks, most software professionals now accept the existence and validity of these tradeoffs but as Brooks himself once ruefully observed, quoting famous maxims is no substitute for managing by them.

With so many unknowns out there, why don’t we make better use of what we do know? Most software “failures” are attributable to the human penchant for unfounded optimism. Under pressure to win business, organizations blithely set aside carefully constructed estimates and ignore sober risk assessments in favor of plans that just happen to match what the company needs to bid to secure new business. Lured by the siren song of the latest tools and methods, it becomes all too easy to elevate future hopes over past experience. This behavior is hardly unique to software development. Recently two economists (Carmen Reinhart and Kenneth Rogoff) cited this tendency to unfounded optimism as one of the primary causes of the 2008 global financial crisis. Their exhaustive study of events leading up to the crash provides powerful evidence that optimism caused both banks and regulators to dismiss centuries-old banking practices. They dubbed this phenomenon the “This Time Is Different” mentality. Citing an extensive database of information gleaned from eight centuries of sovereign financial crises, bank panics, and government defaults, Reinhart and Rogoff illustrate a pattern that should be depressingly familiar to software professionals: without constant reminders of past experiences, our natural optimism bias makes us prone to underestimate risk and overestimate the likelihood of positive outcomes.

The best counter to unfounded optimism is the sobering voice of history, preferably supported by ample empirical evidence. This is where a large historical database can provide valuable perspective on current events. Software development is full of complex, nonlinear tradeoffs between time, effort, and quality. Because these relationships are nonlinear, a 20% reduction in schedule or effort can have vastly different effects at different points along the size spectrum. We know this, but the human mind is poorly equipped to account for non-intuitive exponential relationships on the fly.

Without historical data, estimators must rely on experience or expert judgment when assessing the potential effects of small changes to effort, schedule, or scope on an estimate. They can guess what effect such changes might have, but they cannot empirically prove that a change of the same magnitude may be beneficial in one case but disastrous in another. The presence of an empirical baseline removes much of the uncertainty and subjectivity from the evaluation of management metrics, allowing the estimator to leverage tradeoffs and negotiate more achievable (hence, less risky) project outcomes. One of the most powerful of these project levers is staffing. A recent study of projects from the QSM database used 1060 IT projects completed between 2005 and 2011 to show that small changes to a project’s team size or schedule dramatically affect the final cost and quality. To demonstrate the power of the time/effort tradeoff, projects were divided into two “staffing bins”:

- Projects that used small teams of 4 or fewer FTE staff
- Projects that used large teams of 5 or more FTE staff.

The size bins span the median team size of 4.6, producing roughly equal samples covering the same size range with no overlap in team size. Median team size was 8.5 for the large
team projects and 2.1 for the small team projects, making the ratio of large median to small median staff approximately 4 to 1. The wide range of staffing strategies for projects of the same size is a vivid reminder that team size is highly variable, even for projects of the same size. It stands to reason that managers who add or remove staff from a project need to understand the implications of such decisions.

Regression trends were run through each sample to determine the average Construct & Test effort, schedule, and quality at various points along the size axis. For very small projects (defined as 5000 new and modified source lines of code), using large teams was somewhat effective in reducing schedule. The average reduction was 24% (slightly over a month), but this improved schedule performance carried a hefty price tag: project effort/cost tripled and defect density more than doubled.

For larger projects (defined as 50,000 new and modified source lines of code), the large team strategy shaved only 6% (about 12 days) off the schedule but effort/cost quadrupled and defect density tripled.

<table>
<thead>
<tr>
<th>At 5K ESLOC</th>
<th>Schedule (Months)</th>
<th>Effort (Person Hours)</th>
<th>Defect Density (Defects per K ESLOC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small teams</td>
<td>4.6</td>
<td>1260</td>
<td>3.7</td>
</tr>
<tr>
<td>Large teams</td>
<td>3.5</td>
<td>4210</td>
<td>9.2</td>
</tr>
<tr>
<td>Avg. Difference</td>
<td>-24%</td>
<td>334%</td>
<td>249%</td>
</tr>
<tr>
<td>(Large team strategy)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| At 50K ESLOC        |                   |                       |                                      |
| Small teams         | 7                 | 3130                  | 1.2                                  |
| Large teams         | 6.6               | 13810                 | 3.9                                  |
| Avg. Difference     | -6%               | 441%                  | 325%                                 |
| (Large team strategy)|                   |                       |                                      |

The relative magnitude of tradeoffs between team size and schedule, effort, and quality is easily visible: large teams achieve only modest schedule compression while causing dramatic increases in effort and defect density.
What else can the data tell us about the relationship between team size and other software management metrics? A 2010 study by QSM consultant and metrics analyst Paul Below found an interesting relationship between team size and conventional productivity (defined as effective SLOC per unit of construct and test effort). To make this relationship easier to visualize, Paul stratified a large sample of recently completed IT projects into 4 size quartiles or bins, then broke each size bin into sub-quartiles based on team size. The resulting observations held true across the entire size spectrum:

- In general, productivity **increased with project size**
- **With any given size bin productivity decreased as team size went up.**

To see the relationship between average productivity and project size, compare any four staffing quartiles of the same color in the graph below from left to right as size (bottom or horizontal axis) increases:

As the quartiles increase in size (bottom axis), average productivity (expressed as SLOC per Person Month of effort on the left-hand axis) rises. The slope is reversed for projects of the same size (i.e., within a given size quartile). To see this, compare the four differently colored box plots in the second size quartile highlighted in blue. The size and staffing vs. productivity relationships hold true regardless of which Productivity measure is used: SLOC per Person Month, Function Points per Person Month, and QSM’s PI (or Productivity Index) all increase as project size goes up but decrease as team size relative to project size increases. The implication that the optimal team size is not independent of project scope should not surprise anyone who has ever worked on a project that was over or under staffed but the ability to demonstrate these intuitively sensible relationships between scope and team size with real data is a valuable negotiation tool.

**Determining the Optimal Team Size for your Project**

If the data suggest that optimal team size is related to project scope, it should be able to help us find the right staffing strategy for projects of various sizes. In a study conducted in the spring of 2011, QSM Consultant Don Beckett decided to explore
the best team size for different project sizes and management goals. He divided 1920 IT projects completed since 2000 from
the QSM database into four size bins: less than 4000, 4001 – 9400, 9401-25000, and over 25000 SLOC. For each of these
size bins, he determined median effort (SLOC/PM) and median schedule (SLOC/Month) productivity values. Based on the
results, he assigned projects to one of four categories:

| Better than average for effort & schedule | Worse than average for effort & schedule |
| Better for effort/worse for schedule     | Worse for effort/better for schedule     |

As the chart below shows, projects in the smallest size quartile (under 4000 SLOC) using teams of **3 or fewer people** (blue bars) were the most likely to achieve balanced schedule and cost/effort performance. Teams of **2 or fewer** (purple) achieved the best cost/effort performance and teams of **2-4** (yellow) delivered the best schedule performance. Teams that used **more than 4 people** achieved dramatically worse cost/effort and schedule performance (green bar). This process was repeated for projects in the next 3 size quartiles and the results were entered into a team size matrix:

<table>
<thead>
<tr>
<th>Size Bin</th>
<th>Schedule Optimized</th>
<th>Cost/Effort Optimized</th>
<th>Balanced Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - 4000 ESLOC</td>
<td>2 - 4</td>
<td>2 or fewer</td>
<td>3 or fewer</td>
</tr>
<tr>
<td>4000 - 9400 ESLOC</td>
<td>2 - 6</td>
<td>3 or fewer</td>
<td>3 or fewer</td>
</tr>
<tr>
<td>9401 - 25000 ESLOC</td>
<td>2 - 4</td>
<td>4 or fewer</td>
<td>2 - 4</td>
</tr>
<tr>
<td>Over 25000 ESLOC</td>
<td>4 - 6</td>
<td>5 or fewer</td>
<td>2 - 6</td>
</tr>
<tr>
<td>Large Projects &gt; 70000 ESLOC</td>
<td>10 - 20</td>
<td>10 - 20</td>
<td>10 - 20</td>
</tr>
</tbody>
</table>

Don’s results confirm the findings from our previous two studies: the maximum optimal team size for cost/effort performance increases steadily with project size. The relationship between schedule performance and team size is less clear, with the optimal team size for balanced schedule and performance falling somewhere in the middle.

**Expert Judgement vs. Empiricism**

Regardless of which estimation methods are used in your organization, uncertainty and risk cannot be eliminated and should never be ignored. Recognizing and explicitly accounting for the uncertainties inherent in early software estimates is critical to ensure sound commitments and achievable project plans.

Measures of estimation accuracy that penalize estimators for being “wrong” when dealing with uncertain inputs cloud this fundamental truth and create powerful disincentives to honest measurement. Recording the difference between planned and actual outcomes is better suited to quantifying estimation uncertainty and feeding that information back into future estimates than it is to measuring estimation accuracy.
So how can development organizations counter optimism bias and deliver estimates that are consistent with their proven ability to deliver software? Collecting and analyzing completed project data is one way to demonstrate both an organization’s present capability and the complex relationships between various management metrics. Access to historical data provides empirical support for expert judgments and allows managers to leverage tradeoffs between staffing and cost, quality, schedule and productivity instead of being sandbagged by them.

The ideal historical database will contain your own projects, collected using your organization’s data definitions, standards, and methods but if you haven’t started collecting your own data, industry data offers another way to leverage the experiences of other software professionals. Industry databases typically exhibit more variability than projects collected within a single organization with uniform standards and data definitions, but QSM’s three-plus decades of collecting and analyzing software project metrics have shown that the fundamental relationships between software schedule, effort, size, productivity and reliability unite projects developed and measured over an astonishingly diverse set of methodologies, programming languages, complexity domains and industries.

Software estimators will always have uncertainty to contend with, but having solid data at your fingertips can help you challenge unrealistic expectations, negotiate more effectively, and avoid costly surprises. Effective measurement puts managers in the drivers’ seat. It provides the information they need to negotiate achievable schedules based on their proven ability to deliver software, find the optimal team size for new projects, plan for requirements growth, track progress, and make timely mid-course corrections. The best way to avoid a repeat of history is to harness it.

About the Author

Kate Armel is the Director of Research and Technical Support at Quantitative Software Management, Inc. She has 12 years of experience in technical writing and metrics research and analysis and provides technical and consultative support for Fortune 1000 firms in the areas of software estimation, tracking, and benchmarking. Ms. Armel was the chief editor and a researcher and co-author of the QSM Software Almanac. She also manages the QSM database of over 10,000 completed software projects.

Endnotes

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
- A continually evolving resource for the DoD, Government, Industry and Academia
- Free to use/free to join

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

Current Gold Practices:

- Acquisition Process Improvement
- Architecture-First Approach
- Assess Reuse Risks and Costs
- Binary Quality Gates at the Inch-Pebble Level
- Capture Artifacts in Rigorous, Model-Based Notation
- Commercial Specifications and Standards/Open Systems
- Defect Tracking Against Quality Targets
- Develop and Maintain a Life Cycle Business Case
- Ensure Interoperability
- Formal Inspections
- Formal Risk Management
- Goal-Question-Metric Approach
- Integrated product and Process Development
- Metrics-Based Scheduling
- Model-Based Testing
- Plan for Technology Insertion
- Requirements Management
- Requirements Trade-Off/Negotiations
- Statistical Process Control
- Track Earned Value
An Innovative Approach to Quantifying Uncertainty in Early Lifecycle Cost Estimation

By SEI Cost Estimation Research Group: Robert Ferguson, Dennis Goldenson, James McCurley, Robert Stoddard, and David Zubrow

The inaccuracy of cost estimates for developing major Department of Defense (DoD) systems is well documented, and cost overruns have been a common problem that continues to worsen. Because estimates are now prepared much earlier in the acquisition lifecycle, well before concrete technical information is available, they are subject to greater uncertainty than they have been in the past. Early lifecycle cost estimates are often based on a desired capability rather than a concrete solution. Faced with investment decisions based primarily on capability, several problems are encountered when creating estimates at this early stage:

- **Limited Input Data** — The required system performance, the maturity of the technology for the solution, and the capability of the vendors are not fully understood.
- **Uncertainties in Analogy-Based Estimates** — Most early estimates are based on analogies to existing products. While many factors may be similar, the execution of the program and the technology used as part of the system or to develop it are often different. For example, software product size depends heavily on the implementation technology, and the technology heavily influences development productivity. Size and productivity are key parameters for cost estimation.
- **Challenges in Expert Judgment** — Wide variation in judgment can exist between experts and the confidence in the input that they provide is generally not quantified and unknown.
- **Unknown Technology Readiness** — Technology readiness may not be well-understood, and is likely to be over or under estimated.

An Improved Method for Early Cost Estimation

In 2011 the SEI introduced the QUELCE (Quantifying Uncertainty in Early Cost Estimation) method, an integrative approach for pre-Milestone A cost estimation. The method aims to provide credible and accurate program cost estimates within clearly defined, statistically valid confidence intervals. QUELCE produces intuitive visual representations of the data that explicitly model influential relationships and interdependencies among the drivers on which the estimates depend. Assumptions and constraints underlying the estimates are well documented, which contributes to better management of cost, schedule, and adjustments to program scope as more is learned and conditions change. Documenting the basis of an estimate facilitates updating the estimate during program execution and helps others make informed judgments about estimation accuracy.

The QUELCE method differs from existing methods because it

- uses available information not normally employed for program cost estimation
- provides an explicit, quantified consideration of the uncertainty of the program change drivers
- enables calculation (and re-calculation) of the cost impacts caused by changes that may occur during the program lifecycle
- enhances decision-making through the transparency of the assumptions going into the cost estimate

**Figure 1** shows the flow of information in a typical MDAP Acquisition, with blue boxes added to represent the contributions from the QUELCE method.

How QUELCE Works

QUELCE synthesizes scenario building, Bayesian Belief Network (BBN) modeling, and Monte Carlo simulation into an estimation method that quantifies uncertainties, allows subjective inputs, visually depicts influential relationships among change drivers and outputs, and assists with the explicit description and documentation underlying an estimate. It uses scenario analysis and design structure matrix (DSM) techniques to limit the combinatorial effects of multiple interacting program change drivers to make modeling and analysis more tractable. Representing scenarios as BBNs enables sensitivity analysis, exploration of alternatives, and quantification of uncertainty.
The BBNs and Monte Carlo simulation are then used to predict variability of what become the inputs to existing, commercially available cost estimation methods and tools. As a result, interim and final cost estimates are embedded within clearly defined confidence intervals. The method can be described as a series of eleven steps, summarized in the following sections.\(^1\) Our recent SEI technical report, CMU/SEI-2011-TR-026, elaborates further on the method and its application.

**Step 1: Identify Program Change Drivers**

The identification of program change drivers is best accomplished by the experts who provide programs with information for consideration in cost estimation. Workshops with DoD contractors, domain experts, and former DoD program managers are used to identify drivers that could affect program costs. These experts should consider all aspects of a program that might change (and affect cost) during the program’s lifecycle—particularly given the new information developed during the Technology Development Phase in preparation of Milestone B. The Probability of Program Success (POPS) factors used by the Navy and Air Force can be used to start the discussion.

**Step 2: Identify States of Program Change Drivers**

In the workshops, experts are asked to brainstorm ideas about the status of each program change driver. The specific, assumed state as proposed by the Materiel Solution is identified and labeled as the nominal state. Experts then brainstorm about possible changes in the condition of each driver that may occur during the program lifecycle. The experts identify possible changes that might occur to the nominal state and use their best judgment for the probability that the nominal state will change.

---

\(^1\) This work was originally described on the SEI blog in a two-part series, “A New Approach for Developing Cost Estimates in Software Reliant Systems.” http://blog.sei.cmu.edu/post.cfm/improving-the-accuracy-of-early-cost-estimates-for-software-reliant-systems-first-in-a-two-part-series
Step 3: Identify Cause-and-Effect Relationships for Dependency Matrix

Once the changed condition—referred to as potential driver states—are fully identified, participants subjectively evaluate the cause and effect relationships among the drivers. Expert judgment is applied to rank the causal effects. A matrix is developed that provides the relationship between nominal and dependent states and contains the conditional probability that one will affect the other, but not the impact of the change. This exercise can result in a very large number of program change drivers and states identified for an MDAP.

Step 4: Reduce the Dependency Matrix Using a Design Structure Matrix

Using the Design Structure Matrix (DSM) technique the change drivers can be reduced to an efficient set that has the most potential impact to cost. The DSM technique is a well-established method to reduce complicated dependency structures to a manageable size. An example of a dependency matrix after DSM transformation created during an SEI pilot workshop is provided in Figure 2.

Step 5: Construct a BBN Using the Reduced Matrix

Using the program change drivers derived from Step 4 and their cause and effect relationships established in Step 3, a BBN is constructed. This BBN is a probabilistic model that dynamically represents the drivers and their relationships, as envisioned by the program domain experts. Figure 3 depicts an abbreviated visualization of a BBN, in which the circled nodes represent program change drivers and the arrows represent either cause and effect relationships or leading indicator relationships. This example shows that a change in the Mission and CONOPS driver most likely will cause a change to the Capability Analysis driver, which in turn will likely effect a change in the Key Performance Parameter (KPP) driver and subsequently the Technical Challenge outcome factor. The
three outcome factors (Product Challenge, Project Challenge, and Size Growth) are then used to predict some of the input values for traditional cost estimation models.

**Step 6: Populate BBN Nodes with Conditional Probabilities**

Conditional probabilities are assigned to the nodes (drivers) in the BBN. Each node can assume a variety of states, each of which has an associated likelihood identified by the domain experts. This allows the calculation of outcome distributions on the variables.

**Step 7: Define Scenarios by Altering Program Change Driver Probabilities**

Domain experts use the BBN to define scenarios. The realization of a potential state in a particular node was specified in Step 6, and the cascading impacts to other nodes and the resulting change in the outcome variables were recalculated. Any change in one or more nodes (drivers) constitutes a scenario. Once the experts are satisfied that a sufficient number of scenarios are specified, they use their judgment to rank them for likely impacts to cost. An example scenario created during an SEI pilot workshop is provided in Figure 4.

![Figure 3: Example BBN](image)

![Figure 4: Partial Example of a Scenario With Two Driver Nodes In A Nominal State](image)
Step 8: Select Cost Estimating Relationships or Tools to Generate an Estimate

Parametric cost estimation models for software use a mathematical equation to calculate effort and schedule from estimates of size and a number of parameters. A decision is made as to which cost estimating tool or tools, CERs, or other methods will be used to form the cost estimate. COCOMO II is a well-known estimation tool and is open source. The SEI has so far developed the relationships between BBN-modeled program change drivers and COCOMO, shown in Figure 5. The use of the commercial SEER cost estimating tool is being explored.

Step 9: Obtain Program Estimates Not Computed by the BBN

The Program Office estimates of size and/or other cost model inputs such as productivity are used as the starting point in this step. Often these values are estimated by analogy and aggregation. They are adjusted by applying the distributions calculated by the BBN.

Step 10: For Each Scenario, Run a Monte Carlo Simulation

From each selected scenario in Step 7, use the outcome to parameterize a Monte Carlo simulation. Along with the information from Step 9, this provides probability distributions for adjusting the input factors to the cost estimating models. This also provides explicit confidence levels for the results. Figure 6 shows the simulation results the SEI obtained when modeling a factor (person-months) in three different scenarios.

Step 11: Report Each Scenario Result Independently

Report the final cost estimates for each scenario, including the nominal program plan. The explicit confidence levels and the visibility of all considered program change drivers allows for quick comparisons and future re-calculations. The transparency afforded by the consideration of alternative scenarios enables improved decision making and contingency planning.

Results and Future Research

QUELCE as an approach to early cost estimation is unprecedented in many ways. The SEI spent much of the past year developing and refining the analytical methods used. So far, trials of the earlier steps of the method have been conducted in workshops, and post hoc reviews of previous estimation artifacts were used for later steps. The SEI’s experience and the results achieved thus far suggest that the approach has considerable merit. Feedback about the value of the approach from the participants in workshops and from leaders in estimation research has been very positive.

Empirical validation of QUELCE is ongoing, and the results of this evaluative research will be used to refine the approach and demonstrate its value. Future efforts will benefit from the participation of programs that are willing to provide access to the artifacts developed prior to Milestone A or to use the QUELCE method in upcoming Milestone A estimates. Through these joint efforts, the SEI will evaluate the extent to which the probabilistic methods proposed improve the accuracy and precision of cost estimates for DoD programs.

Conclusion

Extensive cost overruns have been endemic in defense programs for many years. A significant part of the problem is that cost estimates for unprecedented systems must rely
Scenario 1  
(All Cost Drivers set at their historical distributions)  
Median = 1,227 mths  
Range = 2,664 mths  
Upper 95% Limit = 1,854 mths

Scenario 2  
(Three Cost Drivers set at Nominal State only)  
Median = 290 mths  
Range = 1,441 mths  
Upper 95% Limit = 674 mths

Scenario 3  
(Six Cost Drivers set at Nominal State only)  
Median = 152 mths  
Range = 742 mths  
Upper 95% Limit = 389 mths

Figure 6: Simulation Results for Three Scenarios

heavily on expert judgments made under uncertain conditions. QUELCE aims to reduce the adverse effects of that uncertainty. Important program change drivers and the dependencies among them that may not otherwise be considered are made explicit to improve the realism and likely accuracy of the estimates. The basis of an estimate is documented explicitly, which facilitates updating the estimate during program execution and helps others to make informed judgments about their accuracy. Variations in the range of possible states of the program change drivers that may occur under different likely scenarios are explicitly considered. The use of probabilistic methods combining Bayesian belief systems and Monte Carlo simulation will ultimately place the cost estimates within a more defensible range of uncertainty.

References


AN INNOVATIVE APPROACH TO QUANTIFYING UNCERTAINTY IN EARLY LIFECYCLE COST ESTIMATION (CONT.)


About the Authors

Mr. Robert Ferguson is an experienced software developer and project manager who joined the SEI after 30 years in industry. His experience includes applications in real-time flight controls, manufacturing control systems, large databases, and systems integration projects. He has also led process improvement activities at two companies. At the SEI he supports customers with problems in estimation and measurement implementation.

Mr. Ferguson is a Senior Member of IEEE. He has Project Management Professional (PMP) certification from the Project Management Institute (PMI).

Dennis R. Goldenson came to the SEI in 1990 after teaching at Carnegie Mellon University since 1982. He is widely recognized as a leading expert in software engineering measurement and analysis. His work over the years has concentrated on advanced measurement and analytical methods with a focus on the performance and quality outcomes of software and systems engineering practices. In addition to his work on early cost estimation in the presence of uncertainty, other recent work related to early cost estimation done with DoD and contractor organizations includes research on capability and requirements engineering. His current and recent work also includes empirical evaluation of software architecture in cloud computing, the quantitative analysis of textual information, and tools to support collaborative processes. Related interests are in voice of customer methods, modeling and simulation, experimental design, and the visual display of quantitative information.

Jim McCurley is a Senior Member of the Technical Staff for the Software Engineering Measurement and Analysis (SEMA) group within the Software Engineering Institute (SEI). During his 14 years at the SEI, his areas of expertise have included data analysis, statistical modeling, and empirical research methods. For the last several years, he has worked with various DoD agencies involved with the acquisition of large scale systems. From 1999-2005, Jim also worked as a member of the Technical Analysis Team for the CERT Analysis Center. The team developed tools and techniques for the identification and remediation of threats to ultra large scale government and military networks and worked with the DoD JTF-GNO, DoD CERT, NSA and the FBI’s NIPC. Prior to joining the SEI, Jim consulted on projects with several governmental agencies and corporations, including NOAA, EPA, Army Corps of Engineers, the Pennsylvania Governor’s Energy Council and Science Advisory Board, the Battelle Institute, Martin-Marietta Environmental Research, the Western Psychiatric Institute and Clinic, and the University of Pittsburgh Medical Center’s transplant program and Center for Medical Ethics. He began programming computers for mathematical and scientific problem-solving in 1966 and has taught courses in FORTRAN, Pascal, Lisp, APL, and several statistical packages at Carnegie-Mellon University. He currently teaches the Improving Process Performance Using Six Sigma, Designing Products and Processes Using Six Sigma, and Implementing Goal-Driven Measurement courses at the SEI.

Robert Stoddard currently serves as a Senior Member of the Technical Staff within the Carnegie Mellon University federally-funded research and development center known as the Software Engineering Institute (SEI). Robert is responsible for research, Air Force Logistic Center client consulting and the development and delivery of advanced training on cost estimation and software predictive analytics, specifically with regards to CMMI High Maturity process performance baselines and models. Robert continues to promote the integrated use of Six Sigma and CMMI as exemplified in a 2008 published
book entitled “CMMI and Six Sigma: Partners in Process Improvement” by Addison Wesley, and is an invited speaker to a wide variety of annual conferences and symposiums held by the SEI and the American Society for Quality (ASQ). Robert remains active with both the IEEE Reliability Society and the ASQ Software Division. Robert holds certifications with the ASQ on five topics including Six Sigma Black Belt, Reliability Engineering, Quality Engineering, Software Quality Engineering and Quality Audits. Robert became a Motorola-certified Six Sigma Black Belt in 1993 and Motorola-certified Six Sigma Master Black Belt in 2003. Robert earned a B.S in Finance and Accounting from the University of Maine, an M.S. in Systems Management from the University of Southern California, and has completed PhD course work in reliability engineering and engineering management. Prior to joining the SEI in 2005, Robert served for seven years as a Motorola Director of Quality and Distinguished Member of Technical Staff and for 14 years as a Texas Instruments Site Software Quality Manager.

Dave Zubrow is a senior member of the technical staff and manager of the Software Engineering Measurement and Analysis (SEMA) initiative within the Software Engineering Institute (SEI). Prior to joining the SEI, Zubrow served as assistant director of analytic studies for Carnegie Mellon University. He is an SEI certified instructor for courses related to process improvement and measurement and analysis, a member of several editorial boards of professional journals, and active in working with organizations on applications of measurement and analysis for management and improvement purposes. Zubrow is a senior member of ASQ and an ASQ Certified Software Quality Engineer.
This article describes the use of parametric modeling to support independent government cost estimates at the feasibility/budgeting, acquisition, and on-going project lifecycle procurement support. Benefits over alternate approaches (analogy and bottom-up) are defined, and typical budgets for this activity are presented.

Introduction

For most system acquisitions, there are three stages at which there is a need for some form of Independent Government Cost Estimate (IGCE) or other cost/price review. The FAR addresses these requirements primarily in FAR 15\(^1\), although FAR 7\(^2\) provides the high level requirement, stating that Agency Heads are responsible for “Ensuring that the statement of work is closely aligned with performance outcomes and cost estimates.” The three acquisition stages where IGCEs are necessary are:

1. As part of the feasibility or budgeting cycle;
2. As part of the proposal analysis during system acquisition; and
3. As part of the on-going procurements involving scope changes during the system lifecycle.

This article provides an overview of recommended processes that a government cost estimator can follow at all stages, with an emphasis on parametric modeling using domain specific high level objects as proxies for effort and/or cost.

It’s useful here to differentiate between price analysis and cost analysis as defined in FAR 15.404. Price analysis is the process of examining and evaluating a proposed price without evaluating its separate cost elements and proposed profit. Cost analysis is the review and evaluation of any separate cost elements and profit or fee in an offeror’s or contractor’s proposal. To some extent, Price Analysis is closer to the field of Economics while Cost Analysis is closer to the field of Accounting. The primary focus of this article will be on Price Analysis, although we’ll touch on Cost Analysis briefly toward the end.

There are many approaches to IGCE, and some specific nuances that are important, but before we get into those areas we will present a brief overview of parametric modeling.

Parametric Modeling Using High-Level Objects (HLOs)

At the core, there are fundamentally three approaches to estimating:

1. Bottom up\(^3\), in which the project is decomposed to individual components that can be estimated by some other means, often expert judgment for labor.
2. Analogy, in which historic budgets or actual expenditures are used as a basis for current estimates. These numbers are normally adjusted to account for known differences between the historic project and the current project, including as a minimum an allowance for inflation based on the relative timeframes;
3. Parametric, in which models are used to forecast cost based on some representation of project size combined with adjusting factors.

Traditional Project Management guidance is that Bottom up is the most accurate; followed by Analogy; followed by Parametric. Whether this is true is highly dependent on the maturity of the parametric models in the domain you are trying to estimate. For example, in the software domain we have had an opportunity to track budgets versus actuals for more than 12,000 projects that were estimated using the above three approaches. What we have found is quite the opposite.

In the software domain, parametric is the most accurate, followed by analogy, followed by bottom up. The standard deviation of the estimate for parametric estimates is 55% smaller than estimates by analogy and 64% smaller than bottom up estimates. These ratios hold more-or-less true for estimates prepared at any stages of the project lifecycle (budgeting, feasibility, planning). A value that is often more critical, at least when dealing with project portfolios or clusters of estimates for a given scope change, is estimation bias. If you look at a statistically significant sample of estimates (e.g.,
20 projects in a portfolio), and you total up both the estimates and the actuals for that collection, the bias is the difference between the numbers. With parametric estimates and a properly calibrated model, this bias approaches zero (we consistently see it under 5% for large organizations, with a random direction). With estimates by analogy, this number is typically closer to 10%, also with a random bias. But with Bottom Up estimates, this number is typically between 15% and 20% with a bias toward under-estimating. In the remainder of this section we’ll discuss parametric estimation in more detail.

As shown in Figure 1, the core requirements for effective parametric estimation in any domain are relatively simple. Step one in the process is to identify one or more high level objects (HLOs) that have a direct correlation with effort. The HLOs that are appropriate are domain specific, although there is sometimes an overlap. Examples of HLOs include yards of carpet to lay, reports to create, help desk calls to field, or claims to process. In activity based costing (ABC), these would be the cost drivers. HLOs are often assigned a value based on their relative implementation difficulty, thereby allowing them to be totaled into a single numeric value. An example is function points, which are a total of the values for the function point HLOs (EQ, EI, EO, ILF, and EIF). Don’t worry if you’re not familiar with those terms, it’s the idea that they represent something that’s important here.

HLOs may have an assigned complexity or other defining characteristics that cause an adjustment in effort (e.g., simple report versus average report). It’s also typically necessary to have a technique for managing work that involves new development, modifications or extensions of existing components, or testing/validation only of existing components. Various formulas or simplifying assumptions may be used for this purpose. For example, in the case of reuse, the original COCOMO I model reduced the HLO size to:

\[ HLO = HLO \ast (0.4DM + 0.3CM + 0.3IT) \]

where DM is the percent design modification (1% to 100%); CM is the percent code modification (1% to 100%); and IT is the percent integration and test effort (1% to 100%).

Step two is to define adjusting variables that impact either on productivity, or on economies (or diseconomies) of scale. The productivity variables tend to be things like the characteristics of the labor who will be performing the work or the tools they will be working with; characteristics of the products to be created (e.g., quality tolerance) or the project used to create them; and characteristics of the environment in which the work will be performed. The variables that impact on economies or diseconomies of scale are typically things that drive the necessity for communication/coordination, and the efficiency of those activities. These adjusting variables are important both to improve the accuracy of any given estimate, and also to normalize data to support benchmarking across companies or between application areas.

Step three involves defining productivity curves. These are curves that allow a conversion between adjusted HLO sizing counts and resultant effort. They are typically curves (versus lines) because of the economies or diseconomies of scale that are present. Curves may be determined empirically or approximated using industry standard data for similar domains. Curves may also be adjusted based on the degree to which the project is rushed. In any event, procedures are put in place to collect the necessary data to support periodic adjustment of the curves to match observed results, a process called calibration.

The outputs of the process are driven by the needs of the organization. These outputs can be broken down into three major categories:

1. Cost (or effort, which is equivalent for this purpose): In addition to the obvious total value, most organizations are interested in some form of breakdown. Typical breakdowns include breakdowns by organizational unit for budgetary or resource planning purposes; breakdowns by type of money from a GAAP perspective (e.g., opex versus capex); or breakdown by WBS elements in a project plan.
These outputs will also typically include labor needed over time, broken down by labor category. These outputs are generated using a top down allocation.

2. Non-Cost Outputs: Non-cost outputs are quantitative predictions of either intermediate work product size, or non-cost deliverable components. Examples include the number of test cases (perhaps broken down by type), the engineering documents created with page counts, the number of use-case scenarios to be created, or the estimated help desk calls broken down by category. These outputs are typically created using curves similar to the productivity curves, operating either on the HLOs or on the total project effort.

3. Lifecycle Costs: If the estimate is for a product to be created, delivered, and accepted then the cost and non-cost items above would typically cover the period through acceptance. In most cases there would then be an ongoing cost to support and maintain the delivered product throughout its lifecycle. These support costs are relatively predictable both in terms of the support activities that are required and the curves that define the effort involved. For many of them, the effort will be high immediately following acceptance, drop off over the course of one to three years to a low plateau, then climb again as the product nears the end of its design life.

In the next few sections of this article we’ll focus on the application of parametric modeling techniques to IGCEs in support of the three procurement phases.

Feasibility and Budgetary Analysis

The purpose of the Feasibility/Budgetary estimate is to determine the estimated final cost with sufficient accuracy to support accurate prioritization of competing projects and the allocation of sufficient funds in future accounting periods to support the actual project procurement. Problems with estimation accuracy (the standard deviation of the estimate) will distort project prioritization decisions. Problems with estimation bias will result in future funding shortfalls that will create problems for multiple projects as the budgets are re-allocated.

In many ways, these early stage estimates are the most difficult. The Cost Analyst is asked to estimate the cost to build something that is often only understood at the highest level of granularity. A common mistake is to attack the problem by looking for pieces of the puzzle that are well understood, defining those to a high degree of granularity, estimating those, adding some kind of fudge factor for the unknown components, and using that as your estimate. This is wrong for two reasons. First, a lot of time is spent on the well understood components to achieve a high degree of precision, but that precision is lost when the total effort is created. So it wastes time. Second, this approach gives a sense of understanding of the project that is overly confident, thereby failing to adequately compensate for the areas of unknown and resulting in a tendency to underestimate (sometimes badly). Worse yet, the detail in some areas can mask this lack of overall completeness from a reviewer with oversight responsibility.

One alternate approach is to use a parametric modeling approach that makes heavy use of estimation by analogy concepts. The steps involved are as follows:

- Break the problem domain down into components where there is data available for analogous components. This might be the entire product, or a part of the product.
- Use engineering judgment to select where this project lies for that component on a ranked list of the analogous components. Interpolate or extrapolate if necessary.
- Use the HLO count for the analogous component as input to the parametric model for this component.
- Set adjusting variables specific to this project.
- Generate the estimate for this component.
- Repeat for other components, then sum.

Often you’ll use this approach for the labor components needed to create the component (e.g., programming); then use cost estimating relationships (CERs) to create the estimates for other project roles such as project management.

A second approach often works well when you are replacing an existing item.

- Count the size of the existing item using a suitable set of HLOs;
- Adjust the count down using reuse equations if you expect to get any reuse of the design or portions of the existing system;
- Add to the count based on anticipated expansion of capabilities;
- Set adjusting variables specific to this project;
- Estimate the effort.

A third approach works well in an environment where you are primarily making on-going modifications to existing systems.
• Identify the impacted systems;
• Make a guess at the degree of impact (e.g., very small, small, average, large, very large). If you can’t guess, use average;
• Use historic data about HLO equivalent effort for modifications to that system.
• Set adjusting variables specific to this project; and
• Estimate the effort.

In our experience, these approaches can consistently generate estimates within +/- 50% with less than 5% bias, and standard deviations closer to +/- 25% are often achievable for any given organization. Further, the time required to generate the estimates is less than half the time required using other techniques.

**Initial System Procurement**

The objective of proposal analysis is to ensure that the final agreed-to price is fair and reasonable [FAR 15.404-1]. The FAR uses the phrase “fair and reasonable” throughout, although these are actually two separate but related items when applied to price analysis.

A proposal price analysis looks at the reasonableness of a vendor’s proposed price as a risk reduction measure. A reasonable price is one where there is a high probability that the vendor can deliver the solution for that price. An unreasonably low price is a problem for two primary reasons:

- The vendor may not be willing or able to perform the necessary work at a loss, thereby resulting in a situation involving potential litigation, project cancellation, or major scope changes that are unfavorable to the government. All of these options are extremely expensive for the government; and
- The vendor may deliver the product at the price quoted by significantly reducing quality, thereby increasing government work both during the project and during the product lifecycle. Again, this is an expensive option.

While there are unscrupulous vendors who will buy-in to a contract with a low-ball bid, hoping to use change requests to bring the project back into profitability; the more common reason for unreasonable pricing is that the vendor’s understanding of the project scope does not match the government’s understanding of project scope. In that situation in particular, a price reasonable test of the vendor’s proposal is of benefit to both the government and the vendor. Price reasonableness tests are generally pass-fail, and the vendor generally does not have an opportunity to modify their cost proposal if it fails the test.

Price fairness looks at the vendor’s proposed pricing relative to equivalent pricing for the same product offered by others in the industry. In a situation where there are competing bids that are roughly in-line with each other, price fairness is assured through the competitive process. In a situation where there is historic precedence for the quoted price (e.g., catalog price, similar procurements for the same goods and services in the past), then this historic information may be used to confirm price fairness. Where price fairness becomes critical for the cost analyst is the situation where there is only one bid that is under consideration. This might be because only one bid was submitted, or it might be that only one bid offered a solution that meets the technical or administrative requirements of the procurement. In that situation, the approach to determining price fairness is to do a cost (vice price) analysis.

One approach to doing the cost analysis is to look at the vendor’s detailed build-up of costs (level-of-effort, direct salary, fringe rates, overhead, G&A, fee, etc.) and to assess each for reasonableness. The direct salary can be validated using wage and labor charts or through competitive labor market surveys. The fringe rates can be validated through those same surveys. The overhead and G&A rates can be validated against competing companies of a similar size and competition. Fee can be validated against other similar contracts, based on risk and other factors affecting appropriate profit margins. However, the validation of the vendor’s proposed level-of-effort is the difficult part. This is where parametric modeling can offer assistance. The vendor’s proposed solution is analyzed to identify suitable HLO counts and adjusting factors, and the models are then used to forecast effort based on productivity curves from similar historic projects. The resultant modeled effort is then compared with the vendor’s proposed effort to determine if the effort is fair.

Unlike reasonableness, problems with price fairness are often handled through vendor negotiation and submission of a best and final offer. The reason for this difference is as follows. Cost is always one of the evaluation criteria. If a vendor wins the competition with a low price, but then fails the price reasonableness test, and we allow the vendor to raise their price to reasonable levels, then we have given that vendor a competitive advantage over other companies who bid the correct price. One of those companies should be awarded the contract instead of this vendor. On the other hand, if a vendor wins the procurement but fails the fairness
test, then their bid is the most attractive to the government of the submitted bids at their current, high price. Lowering their price through negotiation is in the best interest of the government, but would not change the vendor’s relative ranking among the competition.

The third area where IGCE is used is during the life of a project, and that will be the topic of our next discussion.

**Project Lifecycle Acquisition Support**

While the baseline project scope and price are defined upon contract award, it is not unusual for there to be dozens or hundreds of legitimate scope changes during the course of the contract, each with an impact on cost and schedule that must be analyzed. We’ll discuss this process here. There are two primary sources of scope change: scope ambiguity and requirements changes. We’ll start by discussing what is often the larger of these, scope ambiguity.

**Scope Ambiguity**

Low ambiguity projects are those for which a comparable product exists and to which this product may be compared. A good example is a tract house. If you want to know what your house will look like, go look at the model house. It will look just like that. Other examples of low ambiguity projects are shrink-wrap software installations and hardware installations. Many projects fit into the low ambiguity category, and scope can and should be tightly managed on these projects.

Moderate ambiguity projects are those where the components of the final product all have comparable products elsewhere for comparison, but the combination of those components in this product is unique. Consider a custom built home using standard construction techniques. The cabinets, flooring, windows, etc. are all components that are well defined but uniquely assembled. Even things like flooring can be unambiguously described, as in “100 square yards of carpet with padding.” Some Commercial Off-The-Shelf (COTS) software installations are another example of moderate ambiguity projects. One characteristic of these projects is that professionals in that domain tend to see these as low ambiguity, while customers for which this is their first exposure to this domain will often see them as high ambiguity. For example, a builder might be able to look at a well drawn set of blueprints and understand the scope exactly, but a homeowner might not understand what is being built until they can actually see it. This disconnect in perception can result in friction. Here, effective scope management centers on early education of the customer about exactly what the product will be when delivered, or reducing the customer’s scope ambiguity in other words.

A high ambiguity project is one in which there are no comparable products for direct comparison. An example might be a mission to Mars. A more common example would be virtually every custom software project. Because there are no comparable products for direct comparison, we will have scope ambiguity.

**Dealing with Ambiguity**

For high ambiguity projects, a significant portion of the project effort is spent reducing that ambiguity. Looking at Figure 2, we see that in reality the project doesn’t have a single baseline, but rather it has an evolving baseline that is progressively elaborated. For example, on a software project we start with business need, then define the requirements, then turn those into high level design, turn that into detail design, possibly do a prototype, turn that into physical code, and turn that into delivered functionality (the actual product). At each step in the process we are elaborating on the work of the previous stage, refining, clarifying, correcting, and changing as we go.

**Requirements Change**

The second source of change is actual requirement changes. Here I’m talking about actual changes in the underlying requirements. Perhaps there is a building code change that...
must be met. Perhaps the legislature has changed a law which affects how the software should operate. Perhaps the operational feedback from the field requires modifications in the armor plating. The longer a project lasts, the more you can count on the requirements changing during the project. For software projects, the rate of requirement change varies from 3% to 15% per year, with 12% being a common number. The good news is that these changes are clear candidates for scope management through the change control process. The bad news is that because they are changes to a project that is already underway, in most cases the only person who can implement the change is the company already doing the work. This sets up a sole source bidder situation that is uncomfortable for the customer in the best case, and that can be used by the vendor to generate large profits at government expense in the worst case.

Let's take a DoD project as an example. The approach to managing change requests might look something like Figure 3. A change request is prepared; an impact assessment plan (IAP) or some similar document is prepared; the vendor estimates costs; the government reviews the estimate. If the costs are approved, the government then attempts to justify the additional funding and authorize the change. If they do not agree with the estimate, or they cannot justify the expenditure, they negotiate with the vendor. At this point, there is often a significant amount of stress and finger pointing, with no-one very happy.

The first attempt to fix the problem often looks like Figure 4. Here, the government creates an independent government cost estimate (IGCE) for validation of the vendor estimate. As long as the IGCE estimate matches the vendor estimates, this approach works swimmingly. Unfortunately, when they don’t match, I have yet to meet a vendor that didn’t think their estimate was right and the IGCE estimate was wrong. Again, lots of friction and finger pointing. This time, because of the clear discrepancy between the two estimates, the almost inevitable result is that the entire process gets deadlocked.

Now, let’s look at an alternative that does work consistently, on projects of all sizes and complexities (Figure 5). The first step is that a parametric model based approach to estimation is agreed to by all stakeholders, configured, and validated. This might be a software cost estimating model, a construction cost estimating database, or any other standard estimating...
method that is objective and verifiable. This approach, and the underlying configuration, should be set up and agreed to with no money on the table, thereby allowing stakeholders to be objective. Once it is finalized, then an IAP is completed both the Vendor and an independent government expert review the IAP and use the common, agreed to estimating model to create independent estimates. If they match, then the estimate is approved and the change control board can proceed with a cost benefit review of the change request. If they do not match, then there is a reconciliation meeting to identify where and why the underlying assumptions do not match. The key here is that the discussion is focused on technical aspects of the work to be done (e.g., HLO object definitions), not on costs or efforts. Once that is clarified, then both parties run the models again and compare results. In my experience, 70% of the IAPs result in estimates that match and therefore do not need a reconciliation meeting at all; another 24% are resolved following one reconciliation meeting, and 6% either require another meeting or require that the original change request be clarified as to intent.7

A Real-World Example

Let’s look at an actual example of the application of these techniques. Cost Xpert Group, Inc. (www.costxpert.com) was hired to perform an independent cost analysis for the Transcom DEAMS Increment 2 program, and the resultant work was performed by William Roetzheim. DEAMS is a defense accounting and financial management information system (MIS) that is being implemented using Oracle’s eBusiness product (specifically, Oracle Federal Financials) using a spiral methodology. Increment 1 of the project involved spirals 1 and 2, while increment 2 involves spirals 3 and 4. There will be an Increment 3 procurement that will encompass spiral 5. Because the project is an ERP implementation, RICEW1 was used as the HLO set (Reports-Interfaces-Conversions-Enhancements-Workflows) with COCOMO environmental variables as the productivity adjusting variables. Help desk support was modeled using user counts by type as the HLO with system size in equivalent function points as the adjusting variable. Lifecycle support costs for deployed spirals that must be supported while building the next spiral were modeled using COCOMO II maintenance effort equations. Program management, blueprinting, and IOT&E support were modeled using Cost Estimating Relationships (CERs). The resultant model calculations were compared to actuals following increment 1 and found to be accurate within an 8% tolerance range. For an example of the use of these techniques to manage change requests during the life of a project, see [Tangella, 2010].

Summary

IGCEs are an important part of system acquisition at all stages of the project life, from budgeting through procurement through final delivery. Parametric models are a useful tool to support these estimates, offering improved accuracy and significantly reduced internal bias. IGCEs may be developed by government employees working for the agency involved; developed by an agency wide center of expertise (e.g., Navy Center for Cost Analysis, Air Force Center for Cost Analysis);

---

1 We often add an additional I to the ERP HLO set, covering Installations, although that was not done in this case.
or created by outside experts brought in for this purpose under a consulting agreement. The cost of performing the IGCEs will vary based on the size and complexity of the project; and for Price fair and reasonable analysis, also based on the number of proposals received. For IT projects in the $100M to $1B range, typical budgets would be:

- Feasibility/Budget: $50K to $100K
- Proposal Price Analysis: $100K to $200K
- Lifecycle Acquisition Support: $50K to $100K per year

About the Author

Mr. Roetzheim (PMP, RMP, MBA, CRISC, CISA, IFPUG) is founder and CEO of Level 4 Ventures, Inc., an SDVOSB that offers independent cost analysis, risk management, and project oversight services for large IT projects. He has written 27 published books, over 100 articles, and three columns. He is a frequent lecturer and instructor at multiple technology conferences and two California universities. He can be reached at William@level4ventures.com.

Endnotes

i FAR 15.404-1
ii FAR 7.103
iv For an in-depth look at parametric estimation approaches in support of acquisitions, see: Parametric Estimating Handbook, DCAA, 1996
vi See also OMB Circular A-94, Guidelines and Discount Rates for Benefit-Cost Analysis, 1993
vii For an example of the successful application of this approach on a government contract, see: Tangella, Lakshmi, Validation of Software Cost Estimates, IBM, 2010
Selling Your Software Estimate

By Arlene Minkiewicz

In many instances the software industry continues to flounder when it comes to understanding how to properly plan and manage software development projects. One need not look far to find evidence that software project planning and management is a complicated challenge. Software estimating is indeed a hard thing to do. But often the problem goes deeper than that. There are many practitioners in our industry who have mastered this skill and still we seem to have way too many projects failing. Software projects often fail not because software professionals lack the proper estimation skills but rather because they lack the skills to negotiate successfully with business leaders during the planning phase and throughout the life of the project. This paper discusses the importance of well reasoned and purposeful negotiation in selling an estimate to a business or customer and presents some strategies for planning and executing such a negotiation.

Introduction

Having spent almost 30 years in the software industry, I think I can safely say that we continue to flounder when it comes to understanding how to properly plan and manage software development projects. One need not look far to find evidence that software project planning and management is a complicated challenge. Quantity is illusive when discussing software, making it hard to determine how ‘big’ a software project is likely to be. Even in situations where software projects are properly sized, they have a tendency to grow once underway. Software technology is constantly evolving, complicating translation of past performance into future predictions.

It’s true. Software estimation is a hard thing to do. Despite this there are many practitioners in our industry who have mastered this skill. And yet we still seem to have way too many software projects failing. It is this author’s theory that the estimating challenges cited in the previous paragraph are only part of the problem. Software projects often fail not because software professionals lack the proper estimation skills but rather because they lack the skills to successfully negotiate with business leaders during the planning phase and throughout the life of a project.

Project leaders don’t want to embark on projects that are destined to fail – they want their projects to be successful. On the other hand they do want to perform in ways that make the business and its leaders happy. Sometimes these two goals are in violent conflict because the business needs are greater than the resources that the business chooses to apply to meeting these needs. It is often the job of the project leaders to bring this to the attention of the business. No one likes to have to tell the boss no. It is often easier to capitulate, hope for the best and save the bad news for later. This is generally a bad strategy and often leads to project failures.

Decisions around project planning and the management of an on-going software project should be based on cold hard facts – not solely on what the business wants or expects nor on the project teams best guess. Ideally every project decision represents the results of collaboration between the business and the project leaders to develop a plan that delivers to the business an optimal set of capabilities for the investment the business is willing to make for such capabilities. Unfortunately this is not always the case. While it is generally true that all stakeholders in a project have the greater good of the business as a goal, there is often not consensus on how best to drive to this greater good or even as to what constitutes this greater good. And there are individual agendas to contend with as well.

Collaboration requires negotiation. The business wants a certain set of features or capabilities from the project team. Naturally they desire to minimize the time and cost associated with getting these features while also expecting the features to be of a specified degree of quality. The project team wants to deliver quality features and they want to ensure that they have sufficient time and people to make that happen. They also want to prevent ending up in a ‘death march’ situation. The project team develops a plan that to the best of their knowledge satisfies all parties. When this plan is presented to business leaders, one of three things might happen:

• The business leaders accept the plan as is because there is trust between the business leaders and the project team – most likely based on a track record of good planning and estimation.
The business leaders reject the plan and insist on a specific plan that aligns with their cost and schedule targets. The business leaders feel that the plan is too costly or the schedule is too long. This opens the door for a negotiation.

The focus of this paper is the third case. Strategies are presented to facilitate properly preparing for and executing a negotiation around a project plan and the underlying estimate.

According to BusinessDictionary.com a negotiation is a bargaining (give and take) process between two or more parties (each with its own aims, needs, and viewpoints) seeking to discover a common ground and reach an agreement to settle a matter of mutual concern or resolve a conflict. In our case, negotiation is the process through which the project and the business come to an agreement on what is a reasonable set of capabilities the project team can deliver given the resources and time the business chooses to devote to obtaining those capabilities. It is important to remember that this negotiation is most likely not a one-time event. Software projects are notorious for change. Good project leadership will update the project plan with project changes. Each of these updates may set the stage for additional negotiations. The secret to success with this kind of a negotiation is preparation.

**Develop a good software estimate**

The place to start your strategy is with a good estimate. While this paper is not intended to be a missive on the art and science of estimation, it does deserve a brief mention. Good software estimates are developed through knowledge about software projects in the past and knowledge about the current set of capabilities being estimated. Good software estimates are developed following a consistent documented process.

The software estimation process requires first that the software project be well understood with respect to the functional and non-functional requirements. Once this information is understood, the estimator should look to identify similar projects from their past or their industry. By comparing and contrasting the current project with history, the estimator can make a reasoned assessment of the ‘size’ of the project being estimated.

The next step is to establish context for the estimate. Understand what activities or work elements need to be estimated and which are not considered part of the project costs. If, for example, some activities are handled as an overhead function available across the organization, these activities may not need to be included in the project estimate. Once a size has been established and other project factors identified - such as complexities, information about the project team, schedule constraints, etc. – an effort estimate can be performed within this context. Effort can be estimated using cost estimating relationships (CERs), either ones developed internally or delivered through a commercially or publically available tool. In the absence of CERs an estimate can be performed by analyzing productivity on similar projects, adjusting for any peculiarities posed by other project factors, and applying this adjusted productivity to the size. It is best to use more than one methodology to estimate software projects so that a sanity check can be performed.

The effort estimate should be aligned with schedule constraints to determine that the resources are available in a timely fashion. If schedule changes are necessary, it is important to reevaluate effort in light of any new schedule constraints as schedule can be an important cost driver. Finally, the inputs to the estimating process should be evaluated with respect to the estimator's certainty of their values and this analysis should feed an exercise to determine the risk of accepting a specific effort/schedule pair. Using multiple methods to estimate and understanding the risk associated with the estimate may become critical to successful negotiations.

**Understand the Players**

It is important to enter a negotiation with an understanding of what the business or customer has at stake. What is driving the need for the software project in question? Is it necessary to keep the business afloat or is it a new product attempting to fuel growth in a new or expanded market? Determine whether time to market is a key driver or whether it is more important to have a feature rich offering. Also come armed with an understanding of which features the business considers critical to the success of the software and which ones could work their way onto the ‘nice to have list’ if necessary. The more you know about the business’ expectations and goals for the final product, the better you will be able to offer solutions that optimize satisfaction.

It is also important that the negotiator understand what’s at stake for the leaders engaged in the negotiation – which is not always the same as what’s at stake for the business. Is this someone who has rode herd over more than one unsuccessful project and needs desperately to have a win? Is it someone who is new to the company and wants to make themselves a reputation? Is there a personal angle to this person’s interest in the project – maybe it is their innovation or pet project. Understanding what the negotiator (or negotiating team)
has to win or lose makes you much better equipped to know where the wiggle room might be. It is also good to find out beforehand what the negotiator(s) familiarity with software development projects is as well as his or her perception of their mastery in this area (these might not be the same). If the negotiator is a former software project lead, negotiations might go more smoothly because they understand the issues. Or the fact that it has been some time since they were in the trenches may make them insensitive to the complexities of developing software in the current decade.

Prepare a Strategy

Begin preparation by taking a good hard look at your estimate. Imagine that it is your job to evaluate the estimate and view it through the eyes of the person who is paying for the software. What aspects of the project would you expect to raise eyebrows? Identify these areas and be prepared to explain what it is about them that make them more effort intensive than intuition would suggest. Your understanding of the negotiators software development expertise should help in this exercise. For every potentially questionable or puzzling aspect of your estimate – be prepared with a credible explanation as to why it is the way it is.

Enter negotiations ready to educate the negotiator about the nuances, complexities and risks of your project. Examples of such issues might include:

- Team is using a new tool or technique – make sure they understand that even though they read a study touting 50% productivity increases in agile development projects (this is just an example – the author is not proposing to have evidence of this) this does not mean the first project for which your team applies agile practices will be any more productive than your last project. In fact, expect the opposite until the new tool or technique is institutionalized.
- Team has new members, either new to software development or new to the product or market.
- New technology or market – if the software being proposed is ‘out of the box’ for this particular team they need to understand that this increases the complexity of the project and increases the amount of time and effort needed to implement the software.
- Non functional requirements that push the state of the art. It may be the case that most of the functionality is pretty straightforward but that there are non-functional requirements in the area of security, performance, reliability, etc that will make their implementation more complex.
- Team is planning on including Commercial Off the Shelf functionality – while this generally tends to reduce overall effort and schedule for a project, it does not eliminate effort and schedule and activities related to the selection, preparation and integration of these capabilities, and is an important part of the project plan.

And the list goes on. There are many factors both technical and logistical that drive the costs for your project. Some of them are obvious and don’t require much discussion. It’s important to make sure that the less obvious ones are highlighted and understood. Additionally it should be made clear what, if any risks are inherent in the estimate based on uncertainties in the technology or the estimate itself.

An invaluable tool in negotiations is good historical data. As mentioned earlier, your estimate should be based on data reflecting the historical performance of your team, your organization or your industry. This same data can be used to sell your plan to the business. Having a good visual representation of this historical data – such as a scatter plot of previous projects - which shows your current project on or need the trend line is a great way to get across the point that this estimate was well thought out and informed by history. It may also be useful, though not always advisable depending on the circumstances, to come armed with evidence of past failures of the organization, especially if those failures can be linked to previously unsuccessful negotiations. If multiple methods were used to arrive at an estimate it is good to show how they are similar and be able to explain away any differences in a logical manner. In the absence of organizational data, industry benchmarks are available and should be used to show that your estimate is in an expected range.

Projects need to be performed and delivered under certain constraints. Traditionally, these constraints have been listed as scope, time, cost and quality. They are also used to define the Project Management Triangle, with each side representing a constraint and quality in the center. A passing familiarity with geometry tells us that one side of the triangle cannot be changed without impact on the others.

Understand and respect the Project Management Triangle, and educate your adversaries about it when necessary.
Project managers understand the implications of the Project Management Triangle but still there is often a disconnect. To be more effective, organizations need a framework that respects the triangle and creates a trade space that brokers this negotiation – void of bias, emotion, optimism and personal agenda.

The time constraint refers to the amount of time available to complete a project. The cost constraint refers to the budgeted amount available for the project. The scope constraint refers to the content that must be delivered to ensure the project meets the intended end result. These three constraints are normally competing constraints: increased scope typically means increased time and increased cost; a tight time constraint could mean increased costs and reduced scope; and a tight budget may require an increase in time and a decrease in scope. Quality sits in the middle of the triangle as a reminder that changes in each of the three legs can be made but as these changes are made the area of the triangle will be effected – in other words if you’ve shortened the schedule, even if you’ve made some cost adjustments as well, the quality may not be the same.

The Project Management Triangle is an excellent negotiation tool if it is understood and respected by both sides of the negotiation. A project leader is well advised to come to the negotiation intending to rely on the triangle. In his back pocket should be a set of well thought out cost/schedule/scope trade-offs that can be proffered if necessary. In other words – understand the cost and schedule implications of eliminating or down scoping the features or requirements most likely to be expendable. The ability to speak authoritatively about the various options coupled with visualization around the triangle lends a sense of credibility to the negotiation.

Keep calm and eliminate emotion

Finally, it is important to remain calm and leave emotion at the door. If you believe in the quality of the estimate and have communicated the issues and risks to your leadership, then you’ve done the best you can do. With good historical data to back up your argument and reasonable tradeoffs to present when the cost or schedule is in conflict with organizational goals you should feel comfortable standing firm with your position.

Of course chances are good that you won’t always win, especially if your organization is new to (or completely unfamiliar with) standardized estimating and data collection practices. At some point you may have to back down. At the end of the day the business gets to decide what it wants and when it wants it. It is the project leadership’s job to calmly inform them if and when their expectations are not reasonable and to present them with achievable options. Sometimes the business will respond and sometimes they will fail to bend – believing that this time history will be wrong. In this case the best the project leadership can do is to document their position so that it can be judiciously used in a future negotiation. The institutionalization of good data collection practices should inform negotiations into the future, which are increasingly less painful.

Conclusions

Software estimation is hard but with the right data, tools and methodologies it can be done with a reasonable amount of certainty most of the time. Despite this fact, we see indications everywhere that software projects are continuing to fail. Sometimes the failures are due to poor planning, sometimes they’re due to poor project management and sometimes they come down to the project leader’s inability to sell their plan to the business. Project leaders need to be able to communicate credibly to the business as to the specifics of the project that inform their project plans. They need to come to the table with a plan, the data that supports that plan and a few backup plans if the original plan is unacceptable. They need to educate the business about software development, software estimation and the importance of the Project Management Triangle - using this as a tool to facilitate good negotiations. The best, most well thought out and reasoned software estimate does neither the project nor the business any good if the business won’t accept it, buy into it, and grow it when requirements creep.

About the Author

Arlene Minkiewicz is the Chief Scientist at PRICE Systems L.L.C. In this role she leads the Cost Research activity for the entire suite of cost estimating products that PRICE develops and maintains. Ms. Minkiewicz has over 24 years of experience with PRICE, designing and implementing cost models. Her recent accomplishments include the development of new cost estimating models for software and Information Technology projects. She has published articles on software measurement and estimation in Crosstalk, Software Development and British Software Review. She has received Best Paper awards on several of her recent research projects from two professional estimating societies (ISPA, SCEA), and was named Parametrician of the Year for ISPA in 2002.
The Department of Defense (DoD) and Department of Homeland Security (DHS) need “justifiable evidence and high confidence that acquired systems perform as expected, when expected, are safe, and are secure” [1]. However, the DoD/DHS Software Assurance (SwA) Acquisition Working Group states that “acquisition officials continue to accept software riddled with errors and other security vulnerabilities” [2]. From these statements, it is clear that the U.S. government’s acquisition process for software must change!

This article shows how the U.S government’s software acquisition process can be changed - today - to deliver high quality software satisfying the needs of these government communities.

A critical component of the acquisition process is to ensure that any supplier’s bid for new or maintenance work must conclusively demonstrate, in its ‘pre-award’ proposal, its current ability to perform Inspections compliant with the 2008 IEEE Inspection Standard 1028™-2008 (Section 6) [3] and in accordance with the Federal Acquisition Regulation (FAR) Part 46.2 for Quality Assurance [4].

The current delivered state of software-driven systems must end! The US Taxpayer deserves quality products delivered on-time to meet our nation’s critical needs.

As shown in Figure 1, the supply chains in today’s software acquisition world consist of a wide variety of suppliers spread across the world. Each of these suppliers may have their own standards for development and quality assurance. “Therefore, the responsibility for SwA [software assurance] must be shared not only by software suppliers in the supply chain but also by the acquire in the supply chain who purchase the software” [2]. Additionally, the SwA Acquisition Working Group identified five different acquisition processes - each with their own planning, contracting, monitoring & acceptance [criteria], and follow-on [maintenance] approaches. It is no wonder there are operational problems with delivered software!

The software assurance need begs for an auditable outcome-based acquisition process that, in addition to ensuring high quality, can provide warning of quality and security problems early in the development process – for example when defining contract terms, interface specifications between suppliers, and product requirements.

Since 1991, acquisition professionals have been discussing the concepts, training, roles, and responsibilities that support Performance Based Acquisition (PBA) [5]. Missing is the unity of a coherent acquisition process and availability of relevant standards and tools that tie together all the elements necessary to implement PBA. Outcome-Based Acquisition (OBA) is a specific implementation of PBA and all the elements needed to implement OBA are available today!

**Opportunity**

The Government Auditing Office (GAO) Study of leading companies [6] found that “software developers and acquirers all use three fundamental management strategies to ensure the delivery of high-quality products that are on time and within budget:

1. Working in an evolutionary environment,
2. Following disciplined development processes,
3. Collecting and analyzing meaningful metrics to measure progress.”

![Figure 1: Potential Software Supply Chain Paths](image)
These management strategies also form the basis of the DoD’s Data and Analysis Center for Software (DACS) Gold Standard for Software Acquisition [7].

The availability of 1) the 2008 updated Inspection Standard, 2) Computerized Inspection tools in 2009, 3) the Inspection Compliance Matrix in 2010 along with mitigation of the ten most common Inspection pitfalls in 2007 now provide the capabilities to implement Outcome-Based Acquisition (figure 2).

We will examine these capabilities and see how they converge to provide an auditable and actionable outcome-based acquisition process for delivering high quality products on time and within budget.

1) IEEE Inspection Standard 1028™-2008 (Section 6) [3]

The management strategy of having a disciplined development process – is satisfied by the latest IEEE Inspection Standard which was updated and released in August 2008. The standard specifies formal software Inspections in enhanced and verifiable processes, guidelines, and measurements.

Before continuing, we need to understand the role of Process Assessments (e.g., CMM®, CMMI®). A report of the Defense Science Board [8] observed that “Process Assessments by themselves do not examine the outputs of any development effort and are therefore silent with respect to the quality attributes of any particular product. A positive Process Assessment finding lowers the risk that an organization will produce a low quality product but the [actual] quality of the product itself must be assessed using other methods.”

It is well established that the most effective Product Assessment method available is formal Inspections. Inspections are used to examine work products, such as requirements, design, and code, during development. “The data in support of the quality, cost, and schedule impact of [formal] Inspections is overwhelming. They are an indispensable part of engineering high-quality software” [9]. And formal Inspections are also a DACS Software Acquisition Gold Practice [10]. Inspections complying with the 2008 updated Inspection Standard are the most rigorous and effective means of ‘peer review’, a generic phrase representing a wide variety of review techniques and resulting effectiveness.

IEEE Inspection Standard 1028™-2008 states that the purpose of Inspections is to detect and remove software product anomalies and Inspections are systematic peer examinations that do one or more of the following:

a. Verify that the software product satisfies its specifications
b. Verify that the software product exhibits specified quality attributes
c. Verify that the software product conforms to applicable regulations, standards, guidelines, plans, specifications, and procedures
d. Collect software engineering data (for example, anomaly and effort data)
e. Provide the collected software engineering data that may be used to improve the Inspection process itself and the activities used to produce software products
f. Use the [software engineering] data as input to project management decisions as appropriate
The 2008 Inspection Standard provides comprehensive and detailed criteria against which to measure Inspection execution compliance – a key element of a disciplined development process. However, while formal Inspections have been shown to be highly effective in a disciplined development environment, they must have the proper infrastructure support and upper management’s ongoing adherence to Inspection pitfall prevention techniques to be successfully implemented and sustained. Upper management’s responsibilities for mitigation of Inspection pitfalls can be found in our earlier CrossTalk article [11].

Software Inspections are critical for outcome-based acquisition. Inspections are the best monitoring / “other” method referred to in the Defense Science Board report [8] for ensuring, throughout development, that:

- success is being achieved
- the software will be capable of performing the required functions
- the software is delivered according to schedule, including short-term milestones

A development environment with defined end-state requirements including defined increments accommodates using Inspections to verify requirements, design, system interfaces and other defining material (e.g., Statements of Work). These up-front activities are where the majority of defects (~80%) are historically introduced into a software product [12, 13].

2) Inspection Tools

The management strategy of having meaningful metrics is satisfied by computerized Inspection tools that ensure correct and compliant Inspection implementation of the 2008 Inspection Standard. Inspection tools also provide repeatability, consistency, and outcome-based measurements throughout the development process. The continuous output of an Inspection tool suite during product development is the basis for:

- Ongoing ‘Product’ evaluations with actions identified and resolution status visible
- The auditable outcome-based acquisition process

The characteristics of an Inspection tool suite that is compliant with the 2008 Inspection Standard and generates outcome-based measurements can be found in our earlier CrossTalk article [14] and are summarized in figure 3. Inspection tools are required to aide upper management responsibilities for Inspection planning, execution, monitoring, and result tracking. For example,

Inspection Planning Tools that support ‘what-if’ scenarios, prior to commitment to project Inspections, using industry data or previous project metrics to estimate: the number of inspections needed, hours invested to find and fix defects by Inspection, number and percentage of defects removed by

![Figure 3: Supplier Inspection Tools for Outcome-Based Acquisition](image-url)
Inspection Execution Tools that support Inspection team leaders in determining their team’s readiness to proceed, logging – categorizing – prioritizing defect find & fix status, identifying Inspection process compliance exposures, computing actual hours invested and resulting net savings estimate and return-on-investment (ROI) for each inspection, generating a one-page inspection summary and compliance report for management monitoring.

Inspection Tracking Tools that consolidate and graph inspection metrics for each inspection type (Requirements, Design, Text, Code, Test Fixes); and a consolidated 1-page management dashboard roll-up summary for all Inspections types conducted to-date with supporting graphs.

Project Quality Measurement Tool that provides a running comparison throughout development for achieving Project Quality Plan defect removal goals with the actual defect removals experienced to-date from inspections, testing, and other defect removal capabilities. Inspection tool capabilities also resolve common issues related to why Inspections are not performed:

- many/most Inspection implementations are significantly flawed to the point of the process being discarded
- Inspections are mistakenly felt to be too costly or that they take too long
- Inspections are implemented only on code (not true)
- Inspections are thought to be obsolete in today’s environment (not true)

Figure 4 identifies where Inspection Execution tools are used during an Inspection.

An important aspect of successful use of Inspections is the ability to collect, analyze, and take action based on data (measurements) collected [6]. Inspection tools must collect data at specific points during Inspections to enable management to continuously monitor and assess product development status, track product defect removal against quality plan goals, and quality trends between product components, releases and other projects.

Another benefit of the consistent measurements provided by Inspection tools is the ability for an independent 3rd party Inspection expert or government program office agent to evaluate the results of Inspections by quantifying Inspection benefits and reporting product status for use by either the acquirer’s program manager (e.g., government) or the product supplier’s management.

3) Inspection Compliance Matrix

We have discussed how Inspections can be used in the development process, but how can the Inspection Standard ensure Inspection process compliance by suppliers to produce quality products on schedule? The answer is an Inspection Compliance Matrix which the product acquirer would provide to software supplier candidates for them to complete during pre-contract acquisition bid proposals. The supplier’s completed Inspection compliance matrix would then be evaluated by the acquirer’s Inspection expert to verify the supplier’s current ability to comply with the Inspection.
standard prior to contract award (e.g., generate ongoing actionable Inspection results).

Figure 5 is an example of an Inspection Compliance Matrix provided by the acquirer, to be completed by candidate suppliers.

Of particular importance are the four ‘Recommended (Rec.) Implementation’ columns which identify where the Inspection Standard Actions (shall, may, should) are best addressed by the Supplier’s training, computerized Inspection tools, and Inspection process material. These four columns are populated and provided by the Acquirer for later comparison with how each Supplier candidate conveys their current Inspection capabilities they record in the five adjacent columns under ‘Supplier Implementation’.

Putting it all Together

Figure 6 is an overview of the proposed 4 stage auditable outcome-based acquisition process. Further detail is contained in Table 1. This process would be used to measure progress toward contract award and then successful product delivery.

Table 1 describes each of the steps assigned to the Acquirer and Supplier in the 4 OBA stages in Figure 6. In summary, pre-contract award stages 1, 2, and 3 ensure prospective contract bidders understand Inspections, have the current infrastructure in place to perform standard-compliant inspections, and can demonstrate this prior to contract award. Stage 4 ensures that during post-award contract performance, both the supplier’s management and the government program manager have access to individual and summarized Inspection results throughout product development.

The result of putting the pieces together is an auditable outcome-based acquisition process that can be used TODAY for:

- Inspection Compliance Matrix gap analysis of supplier Inspection capabilities and assessment before contract award
- Proper execution of Inspections throughout development
that comply with the August 2008 IEEE Inspection Standard and Federal Acquisition Regulation (FAR) Part 46.2 for Quality Assurance and Inspections on government contracts.

- Computerized Inspection tool outcome-based measurements allowing supplier and government management, on a continual basis to:
  1. evaluate product quality,
  2. measure development progress,
  3. assess risk

**Conclusion**

“Software vulnerabilities, malicious code, and software that doesn’t function as promised pose a substantial risk to the Nation’s software-intensive critical infrastructure that provide essential information and services to citizens” [2].

The 2008 IEEE Inspection Standard, the Inspection Compliance Matrix, Computerized Inspection tools and Inspection pitfall mitigation techniques along with using an evolutionary development environment approach, ensure correct, consistent, repeatable and sustainable rigorous Inspection execution across distributed environments. Use of these capabilities with their associated benefits (see below) now gives government acquirers, and any other software acquirers along the supply chain (see figure 1), the ability to attain auditable and actionable outcome-based acquisition of products and systems for On-Time Quality capabilities delivered within budget.

**Who benefits from Outcome-Based Acquisition**

**Acquirer/Customer**

Key programs will now receive the capabilities needed when promised with the best quality possible.

**Contracting Officer**

With up to 90% of program cost occurring after deployment, the contracting officer will now contract for the best life-cycle (true) cost of a program.

**Program Manager**

The suppliers current product assessment capability is now a key factor in awarding contracts providing government program managers the ability to monitor ongoing quality, issues, and risk based on the suppliers automated inspection reports.

**Supply Chain Risk Management (SCRM)**

Implementing ongoing outcome-based product assessments (OBPA), the focus of OBA, will reduce/streamline much of the strategies, policies, standards, etc. planned and underway for SCRM

**Suppliers**

Outcome-Based Product Assessments of pre-code activities (e.g., requirements, design) removes defects where 70-80% of defects are typically introduced into products ensuring the best quality at the contracted schedule and cost.

We welcome your comments.
### About the Authors

Roger Stewart is co-founder and Managing Director of the Stewart-Priven Group. He is an experienced Lead Systems Engineer and Program Manager in both government and commercial system development – including Systems Engineering, Software Development, System Integration, System Testing, and Process Improvement.

Previously, Stewart taught the Fagan Defect-Free Process for Michael Fagan Associates (8 years) after spending 30 years with IBM’s Federal Systems Division, (now part of Lockheed-Martin) managing and developing systems for Air Traffic Control, Satellite Command & Control, On-Board Space Shuttle, Light Airborne Multipurpose System (LAMPS Helicopter); and in Commercial Banking, Telecommunication and Networking systems.

Roger has a BS in Mathematics from Cortland University.

---

### Table 1: Key Responsibilities in 4-stage Outcome-Based Acquisition

<table>
<thead>
<tr>
<th>Stage</th>
<th>Step</th>
<th>Acquirer</th>
<th>Supplier Candidate</th>
<th>Acquirer’s Expert</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 1, Pre-Contract Initial Capability Assessment</td>
<td>1</td>
<td>Requires Inspection Standard compliance during acquisition proposal bid response</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Provides the Inspection compliance matrix to Supplier candidates</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Performs gap analysis and maps their current Inspection capabilities to the Inspection compliance matrix, showing their compliance/non-compliance to the “shall”, “may” and “should” in the Inspection Standard</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Assess &amp; Act on Expert’s go/no-go recommendation by eliminating unqualified suppliers</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>Demonstrates Inspection execution compliance to their gap analysis mapping in the Inspection Compliance Matrix</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stage 3, Pre-Contract Execution Assessment</td>
<td>7</td>
<td>Assess &amp; Act on Expert’s go/no-go recommendation by eliminating unqualified suppliers</td>
<td>Demonstrates that Inspection tool reports provide consistent outcome-based actionable &amp; auditable results</td>
<td></td>
</tr>
<tr>
<td>Stage 4, Contract Performance</td>
<td>8</td>
<td>Approve supplier Inspection process, select supplier and award contract</td>
<td>Follow approved Inspection process and provide regular Inspection tool reports for Acquirer’s Program Manager</td>
<td>Monitors Inspection tool reports throughout contract performance and provides periodic product evaluation and inspection process conformance recommendations to Acquirer Program Manager for their ongoing product assessment.</td>
</tr>
</tbody>
</table>
Lew Priven is co-founder and Managing Director of the Stewart-Priven Group. He is an experienced executive with management and technical background in system and software development, software quality training, management development training and human resource management.

Previously, Priven managed the IBM team that developed the inspection process, taught the Fagan Defect-Free Process for Michael Fagan Associates (8 years), and was Vice-President of Engineering & Application Development at General Electric Information Services, Vice President of Application Development for IBM’s Application Systems Division, Director of Operations & Development for the IBM Information Network, Vice President of Information Technology & Human Resources for Satellite Business Systems.

Lew has a BS in Electrical Engineering from Tufts University and an MS in Management from Rensselaer Polytechnic Institute.

The Stewart-Priven Group can be contacted at:
7962 Old Georgetown Road, Suite B, Bethesda MD 20814
Office: 615-754-6685
Web: www.stewart-priven.com
Email: info@stewart-priven.com

References


Senior leaders within DoD, Congress, and industry have grappled for many years with the inability of the Department of Defense’s (DoD) acquisition processes to deliver timely and effective information technology (IT) solutions. The problems are well documented in a number of studies and reports. [Ref 1, 2, 3, 4] Common conclusions of these studies are shown in Table 1.

### Table 1: Problems with DoD IT Acquisition

- Acquisition processes are based on weapon systems and are too slow and costly.
- Requirements and technology change faster than capabilities can be delivered.
- Few IT projects are delivered on time and within budget; many are cancelled before any delivery.

A philosophical turning point in the longstanding discussion about how to fix DoD’s IT acquisition problems was the Defense Science Board’s (DSB) report in 2009. The DSB concluded that it was not possible to tailor the current DoD’s acquisition processes, documented in the so-called 5000 series directives, for acquiring information technology capabilities. Instead, a new acquisition process designed for the special characteristics of information technology was needed [Ref 5]. This philosophy was embraced by Congress in Section 804 of the 2010 National Defense Authorization Act for DoD. Congress directed DoD to develop a new acquisition process for acquiring information technology [Ref 6]. The resulting efforts within the DoD to implement Congressional direction resulted in a strategy for a new acquisition process for IT that was subsequently documented in a December 2010 Report to Congress [Ref 7].

The DoD’s Report to Congress identified top level characteristics of the envisioned new acquisition process. Among these characteristics were the following:

- Short duration projects delivering capabilities in 6-12 months;
- Use of agile processes and development methods;
- Adopting capability-based portfolios as a consistent organizing structure across the disciplines of requirements, funding, and development/procurement (i.e., short duration projects would be specified, funded, and implemented from portfolios);
- Use of pre-tailored acquisition process models (templates) corresponding to types of acquisition;
- Employment of common IT computing platforms;
- Frequent acquisition progress reviews in lieu of infrequent milestone decision points;
- Integrated, concurrent test and evaluation and security certification.

With the publishing of the Report to Congress and the establishment of teams to flesh out the concepts summarized in the Report to Congress, there was optimism in early 2011 that significant improvements were on the horizon. This optimism began to fade in late summer 2011 as progress on DoD’s efforts to further define the new IT acquisition process stalled. Likely, the recommendations to move to a portfolio-based construct across requirements, funding, and program implementation was viewed as having too large of an impact on the various DoD processes. As discussions on implementation of the portfolio construct reached an impasse, work on the other efforts to define a new acquisition process for IT were also stopped.

At present, IT acquisition reform efforts have taken a back seat to DoD’s efforts to adjust to troop withdrawals and budget reductions in 2012 and even larger budget impacts in future years. Nevertheless, improvement in IT acquisition is being addressed as a part of an ongoing revision to DoD’s 5000 series acquisition directives. It is too early to tell if this effort will yield significant improvement in the speed, cost, and effectiveness of IT acquisition.

In view of the lack of progress in achieving significant reform of DoD’s IT acquisition processes, this paper provides a recommended Roadmap for implementation of IT acquisition reforms. In particular, the Roadmap proposes that
DoD prioritize their reform efforts in order to very quickly implement changes that can be done now, in particular changes that can be made without major policy or legislative changes. In parallel, DoD should continue to work on the definition and coordination of those IT reform initiatives that will take longer to implement. In the paper, the characteristics of acquisition reform for IT defined in the December 2010 Report to Congress have been allocated to one of three major Phases of the Roadmap as shown in Table 2.

### Table 2: Acquisition Reform Phases
- Phase 1: Implement Now!
- Phase 2: Change DoD Policies.
- Phase 3: Work Legislation Changes

#### Three-Phased IT Reform:

The major emphasis of this paper will be on Phase 1 of the Roadmap: what can be done now! Phase 1 changes should be implemented immediately and can have a rapid and significant positive impact on the speed and effectiveness of IT acquisition within DoD. Phase 2 of the Roadmap would produce a new DoD IT-specific acquisition policy (i.e., in lieu of DoD 5000) as well as implementation of portfolio management and IT workforce reforms described in the Report to Congress. These initiatives are allocated to Phase 2 because they will require both longer timelines to define and some significant policy changes to implement. In Phase 3 of the Roadmap, initiatives that require modification to law or Congressional processes would be addressed. These include eliminating overly constraining rules for so-called major automated information systems (MAIS) so that they can benefit from processes tailored for information technology. Another initiative in Phase 3 would be changing the way Congress funds information technology, in particular allocating funding to mission areas (portfolios) rather than specific programs. A third focus in Phase 3 would be to implement a single appropriation for IT eliminating the need to “lock in” the implementation approach (i.e., build, lease, or purchase) before appropriate analysis has been completed.

The remainder of this paper will address the five initiatives recommended for Phase 1 of the Roadmap—the activities that should be done immediately. In short, each of the Phase 1 initiatives in the five-point program shown in Table 3 will dramatically improve the efficiency and effectiveness of the DoD IT acquisition processes. Moreover, each of five points can be implemented without policy and legislative changes. The major challenges to implementing these points are the determination that DoD must make rapid changes and overcoming the cultural and bureaucratic inertia.

#### Table 3: Implement Five Point Program—Now!
1. Mandate short-duration, agile projects
2. Require an agile contracting process
3. Employ common platforms and transition to “cloud”.
4. Integrate/streamline T&E and C&A
5. Use acquisition templates

The current fiscal pressures within DoD provide an additional impetus for adopting the five-point plan. It is conservatively estimated that implementing the five points would result in a 10% reduction in the cost of DoD IT acquisition (i.e., on average, reduce the cost of every DoD IT program by 10%). Recognizing that over 50% of the DoD’s $38 billion IT budget goes to contracted services, the potential savings through implementation of the five-point plan are estimated to be about $2 billion dollars per year—with the added benefits of reduced delivery times and IT solutions that are more responsive to user needs! These savings estimates do not include the substantial investments in DoD IT acquisition programs that never produce any results. Implementation of the five-points will eliminate the risk that DoD will need to invest several years and hundreds of millions of dollars before it is determined that the IT acquisition effort should be cancelled. This provocative statement provides a good lead in to the first of the five points.

#### Point 1: Break programs into ~12-18 month projects

The primary objective of Point 1 is to reduce the risk of IT acquisition efforts by very consciously forcing rapid delivery of useful capabilities. Studies have repeatedly shown that smaller projects have a much higher chance of success [Ref 8]. Despite this knowledge, DoD continues to use acquisition processes for IT capabilities that encourage larger programs and extended development or procurement timelines. The solution is straightforward: mandate that IT projects deliver usable capabilities in 12 months or less. In addition to reducing risk, shorter timelines for providing capabilities to users permits timely user feedback that can be incorporated into successive deliveries helping to ensure capabilities are matching user needs. The Office of the Joint Chiefs of Staff has recognized this concept in their recent establishment of the “IT Box” construct for defining requirements [Ref 9]. In essence, the
IT Box forces requirements to be constrained to fit within a “time-box”, a set calendar window of development time, so that they can be developed and fielded rapidly.

In order to implement short-duration projects, larger requirements must be divided so that sub portions can be developed and fielded quickly. Agile development methods ensure continuous communication between developers and users minimizing risk that user needs are not understood by the developers [Ref 10]. Use of common IT platforms (see Point 3 below) permits development efforts to address mission capabilities immediately rather than spending months or perhaps years developing a custom infrastructure platform to meet a user requirement. The recommended metric and action for this point is the following: ruthlessly cancel IT projects that do not deliver capabilities useful to the user community within 12 months (the goal is 6 months or less).

Many will assess the 12-month metric and what might be perceived as a draconian recommended action and conclude that this guideline just cannot apply to DoD. They will argue that DoD is too big and its IT solutions too complex to mandate delivery within 12 months for all IT projects. I note in particular that those involved in business enterprise resources planning (ERP) efforts appear to have grown comfortable with fielding timelines that rival those of major weapons programs, often taking a decade or more before delivery of useful capabilities (or in many cases a decision to cancel the program).

It is the case that the complex IT needs of DoD will most often require solutions that are large and complex, therefore necessitating that larger requirements be divided into multiple, perhaps many, short duration projects. DoD’s system engineers must be directed to determine how to best subdivide larger solutions into “bite-sized” pieces. The response from a program manager or engineer that it cannot be done is false and must be rejected.

In order to develop larger IT solutions through a number of short-duration, quick-delivery projects, there is a need to focus significant attention on how to ensure compatibility and interoperability among the separately-developed IT capabilities. This requires employment of formal engineering disciplines such as architectures, standards, iterative testing, and continuous integration. While these engineering disciplines are often employed in DoD IT acquisition efforts, they are most often used within a monolithic program rather than across many projects that may be developed independently. The additional consequence of the current approach is so-called “stove pipe” IT solutions that cannot be effectively integrated into the larger DoD system of systems context.

Fortunately, there are examples of DoD organizations programs that have successfully implementing complex IT efforts with short-duration projects that deliver capabilities to users very quickly. The team that compiled the Report to Congress identified a number of examples in each of the military services, including DISA, TRANSCOM and DLA. In these examples, short duration projects were effective in delivering incremental capabilities to users in less than 12 months. Success in these examples necessitated that the organizations mature the engineering disciplines necessary to ensure the effective interoperation of the separately-developed, incremental projects. Moreover, each of these projects have shown dramatically reduced risk and increased user satisfaction.

DoD’s challenge with implementing short duration, agile projects is twofold. First, DoD must overcome the current cultural comfort with IT programs taking multiple years to deliver capabilities. Unfortunately, many in DoD sincerely believe that it must take years to deliver IT capabilities! Second, and perhaps more significantly, the common practice of “thorough” implementation of DoD 5000 process steps (i.e., not taking advantage of tailoring opportunities) influences acquisition programs toward larger increments and less rapid delivery. Program managers often start with the objective of rapid fielding only to be consumed by the “mandated” documentation and well-intended oversight activities that quickly erase any hopes of rapid fielding. Moreover, it is not just the design program analysis and software development efforts that must be done more quickly to achieve the 12 month target. Contracting, testing, and project oversight decisions must align with the objective of 12 month (or less) deliveries. Contracting and test are separately addressed in Points 2 and 4.

It is useful to conclude the discussion on Point 1 with the observation made in the opening sentence—shorter duration projects are less risky and therefore can and should be done with less oversight and documentation. Rather than using surrogates such as documents to assess progress, with rapid capability delivery the end user provides direct, high-fidelity feedback by actually using the capability. Based on user feedback, successive projects can be modified, cancelled, or accelerated. A DoD mandate to field IT projects in 12 months or less with penalty of project cancellation will provide the motivation to overcome resistance to change. In addition, this mandate must force each of the critical
processes supporting capability delivery (oversight, design, development, test, contracting, etc.) to operate on the same expedited timelines.

Point 2: Rapidly award contracts

There is abundant evidence that processes used for awarding DoD IT contracts are not aligned with rapid delivery. While the DoD procurement regulations endorse rapid contracting methods and emphasize using risk to guide contracting efforts, contracting often becomes the pacing item for IT efforts. When I was a Program Executive Officer (PEO) in DoD, I pushed the envelope to make contracting more rapid and less costly to government and to contractors. I found that rapid contracting can be done, but it usually required the use of guidelines to the program office and contracting staff. These guidelines would focus on the mission/business benefits of a rapid contracting process. Several guidelines that can assist in implementing an agile contracting process for IT contracting are described in the following paragraphs. Again, these recommendations are focused only on what can be done quickly without changing policy or legislation.

Awarding contracts based on mission areas (e.g., command and control, logistics, IT network infrastructure, etc.) rather than based on specific programs or projects results in contracts being available when a specific project is to be undertaken within the mission area. Often this contract is an indefinite quantity indefinite delivery (IDIQ) type of contract. By focusing the contracts on mission areas, the contract holders have been prequalified in the mission area. When a specific project is to be undertaken, a contract (task) can quickly be awarded to a single contract holder or rapidly competed among multiple mission area contract holders.

There are many IDIQ contract vehicles within DoD and across the Federal government that can be used for development and delivery of IT capabilities. The primary limitation with most of these vehicles is that the time to award a specific contract task order is inconsistent with rapid delivery. Rather than a very streamlined contracting action that takes days or perhaps a week for a short duration (i.e., 12 months or less) project, many task orders take months to award, with attendant high costs to both the government and contractors and with negative impacts to user missions. There is no reason that this process needs to take months. The recommended guideline is that all task order awards be “time boxed” with established maximum timeframes and with no task order contracting process taking more than 21 calendar days from announcement of opportunity to task award. The goal for awarding a task order for a 12 month (or less) effort should be 14 days.

In cases where an IDIQ contract is not appropriate, or for the award of IDIQ vehicles, a similar time box approach should be employed. In this case the guideline is that the time from RFP release to award must be no longer than 120 days. Having implemented this process repeatedly as a PEO, I am confident that it can be done. The challenge is to force the awareness of time to meet user needs as a major factor in all contracting decisions. As a PEO, I negotiated with my program managers aggressive contract award dates prior to RFP release and required the program manager to prioritize tasks to meet the target award dates—it works! The key to a rapid source selection is that activities and tasks must be formally and ruthlessly prioritized in order to focus only on the discriminating items that will permit selection of the “best” contractor for the short-duration project or a series of short-duration projects. DoD budgets and DoD users cannot afford protracted contracting processes that do not focus only on what is most important within the established time constraints.

It is useful to note that once contracting actions can be done rapidly several positive benefits are achieved. It is possible to terminate a contractor for weak performance without causing significant delays in meeting the needs of the end user customers. It is possible to get high quality government personnel to participate in source selection evaluations. Most importantly, the collective resources, both financial and intellectual, of both government and industry become properly focused on quickly delivering the best capability to the end user rather than focusing these resources disproportionately on proposals and evaluations that yield no benefit to the end user and have become increasingly more costly to taxpayers who largely foot the bill.

As a final recommendation for Point 2, all program/project managers and contracting personnel should be required to demonstrate familiarity with the OMB “Myth Busters” Memorandum published in February, 2011 [Ref 11]. OMB’s guidance contains a number of clarifications to common myths that have increasingly resulted in lengthy, more costly, and less effective government contracting.

Point 3: Mandate all programs/projects use established standards platforms

As with DoD weapon systems, current IT acquisition processes require that each program have its own set
requirements, its own funding, and a separate acquisition program office. As such, each IT program is viewed as largely autonomous. This independent structure and the strong culture that has grown up around this structural convention discourages IT programs from being dependent on capabilities outside of the program office’s control. In the past, this gave rise to each program delivering unique, largely redundant and tremendously costly network and computing infrastructures. While separate networks are largely a thing of the past, most large programs still provide their own unique computing hardware, system software including so-called middleware, and user interface capabilities. DoD cannot afford to continue this practice. It is expensive and not effective for meeting DoD user’s needs.

The negative result of each program developing and deploying unique computing hardware and software platforms is truly massive costing DoD hundreds of millions of dollars each year. Most ironic is the fact that these unique platform configurations are each composed of commercially-available off the shelf (COTS) products, albeit uniquely configured for a particular acquisition program. Analyzing, configuring, testing, certifying, and deploying a unique platform for a specific program takes a lot of time, often a year or more. During this time, the program is not delivering the mission capabilities that the program has been charged to deliver. The time and cost of building a unique platform is a total waste of taxpayer money. Moreover, once a unique platform is fielded, this unique configuration significantly drives up life cycle support costs. It simply costs a lot more to support a variety of configurations due to lack of economies of scale in purchasing hardware and software, suboptimal utilization of the platform resources, and the need for additional staff to maintain the unique platform.

A few years ago, I convened three DoD IT acquisition program offices and their contractor support teams to investigate why they were each developing unique hardware and software platforms. In this case, all three contractor teams were from the same company. Each program team was developing a unique “services oriented architecture (SOA)” for their program. Perhaps not surprisingly, each program team was very proud of what they had accomplished and had plans to complete their SOA platform over the next 12 to 24 months. When asked why all three programs could not use a common SOA platform, the chief architect for one program quickly boasted that “my program is a real-time space application”, thereby implying that the other program platforms would not be capable of supporting the space application. After further examination, it became clear that none of the COTS components selected for the “space SOA” were selected specifically because of the (near) real time nature of the application. At the end of the discussion, it became painfully clear to the government and the contractors assembled that the only reason that this one company was spending many millions of dollars and years developing multiple platforms was that the government program offices had asked for a unique program platform as a contract requirement. This was truly a sad observation. Even sadder, however, is that the practice persists today.

The bottom line is that there is no need for unique platforms for 99.9% of DoD IT applications. Unfortunately, the structure and culture of DoD’s acquisition community encourages unique platforms. This must stop. The current model is driving up IT development and support costs and significantly delaying delivery of capabilities to DoD users. What is needed is ruthless standardization of IT infrastructure and mandatory use of standard platforms by all IT programs. DoD already has a number of platforms from which to choose as initial standard platforms. The challenge is not a technical one, it is a cultural one. It is recommended that DoD quickly establish a set of standard platforms and mandate migration to this supported set of standard platforms.

The desire for use of standard platforms aligns well with the desire to take advantage of IT capabilities being provided as a service through what is called a “cloud” model. A cloud is simply a collection of IT resources that can be used on an as-needed basis. Cloud services can include platform services (PaaS) and software service (SaaS) in addition to hardware or data, or applications as a service. For DoD, it makes sense to move to a cloud environment which can reduce the cost of unique services as well as to improve standardization and interoperability. Fortunately, each of the military services, DISA, and many of the other DoD agencies are large enough that they can offer their own cloud capabilities provided either in DoD facilities or in commercial facilities and achieve world class economic results. It is recommended that DoD mandate use of appropriately secured cloud environments (i.e., a ‘cloud first’ policy).

Point 4: Integrate and streamline T&E and C&A processes

The processes used for testing and for certifying the security of IT systems often becomes the “long pole” in getting needed IT capabilities in the hands of DoD users. Clearly, it is important to test capabilities before placing them in the
field and the growing security threats necessitate a strong focus on cyber security. However, the testing and security certification processes typically employed for IT capabilities fail to recognize the realities summarized in Table 4.

Table 4: Realities of IT Test and Security Certification

- Users make the best testers
- Manual security assessments are ineffective
- Security is best evaluated when a capability is operational

Agile development methods recognize the importance of continuous user involvement in the testing and evaluation processes for IT capabilities. Unfortunately, DoD’s acquisition processes which were developed for weapon systems typically require professional testers to perform operational testing. If end users are effectively involved in the development and testing of IT capabilities, the marginal benefits of additional testing by independent professional testers is small and the time delay in getting capabilities to the users becomes unacceptable. A more prudent approach is to rapidly deploy capabilities that have been subjected to testing by actual end users (perhaps through virtual connectivity to the development team) in an incremental manner. This can be achieved by starting with a small subset of actual users and expanding the user community after confidence has been achieved through the smaller deployment. Testing by professional testers should be performed after substantial capabilities have been fielded and are in use. This professional testing effort would be to provide a formal assessment of the effectiveness of the fielded capabilities against the user requirements and to provide recommendations for future development efforts (i.e., modify, expand, or cancel).

Utilization of formal methods and automated tools for security certification is essential. Security certification done by a review of documents only is of almost no value. The commonly-used approach of doing security certification after a system is fully developed is also ineffective and costly in time and money. After an IT capability has been developed, the certifiers often assume the philosophy of “how can we improve security” rather than “what security controls are required to achieve an adequate risk posture”. The result is a cycle of recommendations by the certifiers and negotiations with the developers with delays in fielding and increased costs due to scrap and rework as additional security controls are added after the fact (most often in a suboptimal manner). Instead of addressing security after an IT capability is completed, certification assessments and concurrence by the accrediting authority must be done incrementally and in parallel with the IT development process. The requirement should be that once an IT capability has completed operational testing by users, certification must also be complete.

A final recommendation in this area follows from the fact that security is best assessed in an operational environment. The reason for this is that security threats continue to evolve and IT systems are constantly changing as a result of interactions in the operational environment. Therefore, it is recommended that the majority of security focus for IT capabilities should be applied after operational deployment and using continuous monitoring with automated tools. The practice of focusing most security resources prior to system deployment or in periodic paper based reviews is ineffective and should be abandoned immediately. NIST Special Publication 800-37 [Ref 12] encourages the use of automated monitoring, and provides that it can be the basis for periodic security recertification.

Point 5: Employ acquisition templates

The final point of Phase 1 of the Roadmap is the recommended adoption of tailored templates that can guide program offices as they are strive to tailor DoD’s 5000 Directives to suit information technology acquisition programs. The templates provide a point of departure for IT program offices to quickly determine how their program should be structured. Four templates have been developed under DoD sponsorship. These templates were summarized and endorsed in the December 2010 Report to Congress and align with the major types of IT acquisitions:

1. Application Software Development
2. Commercial Off the Shelf (COTS) Capabilities
3. Integrated COTS
4. IT Services

As an example, the templates reflect the fact that an acquisition program that is to develop application software (on a standard platform) has different activities, risk areas, and indicators of progress when compared to the procurement of a COTS solution. However, experience has shown that by following DoD 5000 processes, DoD program offices typically migrate from a COTS solution approach to a developed solution approach. This should not be surprising; DoD 5000
is targeted for weapon system development. Templates help
to avoid this mistake.

A quick review of the history of programs such as Net-Centric
Enterprise Services (NCES) or the Navy-Marine Corps Internet
(NMCI) shows what happens when IT programs attempt to use a
DoD 5000-based approach to acquire commercial IT capabilities. In the case of NCES, the
objective of rapidly procuring COTS software products evolved
into a lengthy program to acquire and then modify COTS
products. Likewise, NMCI was treated, not as an albeit large
contract for commercial services, but as a developed program
that required formal acquisition milestones, independent
OT&E, and other acquisition processes appropriate for a
developmental program. In both instances, the envisioned
acquisition of commercial products and services evolved into
developmental programs significantly delaying fielding of
important capabilities and dramatically increasing risk and
cost. While NMCI and NCES may be some of the most
visible examples of inappropriate or failure to tailor DoD
acquisition processes, there are many other examples where
well-intended program offices and oversight bodies have
failed to recognize the unique needs of different types of IT
acquisition programs.

The specific characteristics of the templates are only described
generally in this paper. The reader is recommended to review

_A New Model for Acquiring Government Information Technology_[Ref 13] for additional insight into the recommended
templates.

Figure 1 contrasts a few of the top-level characteristics and
processes for the Application Software Development Template and
the COTS Template. It should be noted that this abbreviated
comparison does not show the iterative nature of the
Development and Demonstration (in the case of Application
Software Development Template) and Demonstration and
Deployment (in the case of the COTS Template). It can
be seen that the tasks prior to Milestone B are significantly
different, with the COTS Template focusing on assessing the
commercial marketplace rather than on prototypes and the
Milestone B decision being a procurement decision rather
than a build decision in the case of the Application Software
Development Template.

Key elements of the Application Software Development Template
are as follows:

- Participation by the end user throughout the process.
- Recognizing the evolving nature of IT requirements,
significantly streamlined predevelopment analysis,
prototyping, and documentation focused on the initial
increment (rather than on the full scope of requirements).
- Close linkage of risk reduction/prototyping and

---

<table>
<thead>
<tr>
<th>Application Software Development Template</th>
<th>COTS Template</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ICD</strong></td>
<td><strong>ICD</strong></td>
</tr>
<tr>
<td>Business Case Analysis</td>
<td>Business Case Analysis</td>
</tr>
<tr>
<td>Architectural Alignment/Development</td>
<td>Architectural Alignment</td>
</tr>
<tr>
<td>Prototypes/Risk Reduction</td>
<td>Commercial Market Analysis</td>
</tr>
<tr>
<td><strong>Development &amp; Demonstration</strong></td>
<td><strong>Demonstration &amp; Deployment</strong></td>
</tr>
<tr>
<td>Iteration 1</td>
<td>Iteration 1</td>
</tr>
<tr>
<td>Iteration 2</td>
<td>Iteration 2</td>
</tr>
<tr>
<td>Iteration “N”</td>
<td>Iteration “N”</td>
</tr>
<tr>
<td>Operations And Support</td>
<td>Operations And Support</td>
</tr>
<tr>
<td>Release 1</td>
<td>Release 1</td>
</tr>
<tr>
<td>Continuous and Integrated DT/OT 6 to 18 Months</td>
<td>Integrated DT/OT 1 to 12 Months</td>
</tr>
<tr>
<td>DoD user involvement</td>
<td>DoD user involvement</td>
</tr>
<tr>
<td>Weeks to Months</td>
<td>Weeks to Months</td>
</tr>
</tbody>
</table>

---

Figure 1: Comparison of Application Software and COTS Templates
incremental development such that there is no loss of continuity and no break in progress around the Milestone B decision.

- Rapid, iterative increments of software development (12-18 months) with cost, schedule and performance characteristics assigned as each increment is undertaken.
- Fielding of (agile) software iterations in weeks or months as they are approved by users during development.
- Use of frequent in-process reviews (IPRs) to monitor status and adjust development if necessary.

Key elements of the COTS Template include the following:

- Recognition that the objective is procurement of a product rather than development.
- After procurement, the unmodified COTS product is rapidly demonstrated and then deployed to end users.
- If the user requirements cannot be adequately satisfied with (unmodified) COTS solutions, the program is stopped rather than evolving into a development program.

The third template, Integrated COTS, addresses the fact that DoD often procures multiple COTS products that have been proven individually, but never configured as an integrated capability. This template, therefore, addresses the need for integration of the COTS components and incremental fielding of the integrated COTS capability prior to a full fielding decision. The fourth template, IT Services, recognizes the unique need to define service levels based on market research and customer analysis as well as the importance of focusing on ways after contract award to ensure a continued reduction in the cost of services through employment of evolving technology and process improvement.

Individual programs and projects may also require a combination of templates. For example, enterprise resource planning efforts (ERP) often have a procurement acquisition component for a COTS ERP package (i.e., appropriate for the COTS Template) as well as either an applications software acquisition component (i.e., employing the Application Software Development Template) or a need to integrate several software packages (i.e., using the Integrated COTS Template is appropriate). Using the templates as a point of departure, a program or project office can quickly adapt them to suit the specific needs of an individual program or project.

Summary

Despite the relatively slow progress on a comprehensive restructuring of DoD IT acquisition as envisioned in Section 804 of the 2010 NDAA and the subsequent Report to Congress, there are a number of major improvements that can be made quickly and easily. The five points recommended for immediate implementation will have a significant impact on improving IT acquisition efficiency and effectiveness. Moreover, implementation of these five points does not require rewriting of DoD 5000 or other major DoD policies which can take a year. Rather, they can be implemented quickly with simple clarifying guidelines, one-page policies, and, most importantly, visible emphasis by DoD leadership that the status quo is no longer acceptable.

There is no claim that the five points reflect entirely new thought. Clearly, there has been discussion about shorter development cycles, more agile contracting, and standard IT platforms for years. Likewise the concepts of tailored templates and streamlined test and certification have been proposed before. The five points do, however, reflect a small subset of the concepts proposed in the Report to Congress and in other documents. That is, they provide focus! Moreover, the primary observation in proposing these five points is that implementing just this small set of actions can have an enormous impact on reducing IT acquisition costs and improving the timelines and quality of the products delivered to DoD customers.

The primary challenge therefore is to the DoD leaders who are encouraged to embrace the five points and to quickly act on them as a first phase of DoD’s IT acquisition reform effort. It is not recommended that other elements of the broader IT acquisition reform vision be abandoned. Rather, they should be undertaken in a phased manner to permit achievement of some significant progress in the near term while continuing pursuit of the full reform agenda.

End Notes


### About the Author

**John M. Gilligan** is president of the Gilligan Group, Inc., an IT consulting firm. Prior to his current position he was a senior vice president and director, Defense Sector, at SRA International, Inc. Mr. Gilligan has over 25 years of managerial experience in leading large information technology organizations. He has expertise in acquisition, organizational innovation, program implementation, and IT Security. Recently, Mr. Gilligan served on the Obama-Biden Transition Team focusing on IT within the DoD and Intelligence Communities. Mr. Gilligan has served as the chief information officer for the United States Air Force and the U.S. Department of Energy. He was program executive officer for battle management and command and control for the Air Force. He was a member of the Cyber Security Commission (formed to advise the 44th President). He serves as Chairman of the boards of directors for the Center for Internet Security and for HDT Global, Inc. He is also on the boards of Schafer Corporation, the Software Engineering Institute, and the Armed Forces Communications and Electronics Association. Mr. Gilligan has been a recipient of the Joint Chiefs of Staff Distinguished Civilian Service Medal, Distinguished Executive Presidential Rank Award, Meritorious Executive Presidential Rank Award, and Computerworld’s Premier 100 IT Leaders to name a few. He earned an M.S.E. in Computer Engineering from Case Western Reserve University and an M.B.A. in finance from Virginia Tech University.
Introducing...
...the newest member of the Quanterion Automated Reliability Toolkit family...

Quanterion Automated Reliability Toolkit

<table>
<thead>
<tr>
<th>Feature</th>
<th>QUART</th>
<th>QUART PRO</th>
<th>QUART ER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reliability Advisor</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Reliability Program Cost</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Reliability Improvement Cost</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Warranty Cost</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Statistical Distributions</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Checklists</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Definitions</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Acronyms</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Reliability Potential</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Reliability Allocation</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Reliability Tailoring</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Reliability Approach Assessment</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Derating</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Thermal Design</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Failure Modes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>FMECA Worksheets</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Material Durability Improvement</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Parts Count Analysis</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Software Reliability Prediction</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Redundancy Modeling</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Reliability Adjustment Factors</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Interference Stress Strength Analysis</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Availability Calculator</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Benefit of Reliability</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Reliability Growth Testing - Duane Method</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Reliability Growth Testing - AMSAA-Crow Method</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Reliability Growth Testing - Crow Extended Method</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Reliability Demonstration Test Equal Risk RDT</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Reliability Demonstration Test Variable Risk RDT</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Reliability Test Cost</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Accelerated Reliability Test</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>WeiBayes Substantiation Testing</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Test Results / Confidence Exponential Distribution</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Test Results / Confidence Binomial Distribution</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Design of Experiments 2 Factor DOE</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Design of Experiments 3 Factor DOE</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Design of Experiments 4 Factor DOE</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HASS - ESS</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HASA</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Sampling Plans</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Process Capability</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Weibull Analysis</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Optimal Replacement Interval</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Sparing Analysis (Graphical)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Spares Analysis (Tabular)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

❯❯❯ With over two dozen new or improved tools!

Download for only $399

Quanterion Solutions in a team member of the operation of the Reliability Information Analysis Center (RIAC)

Data & Analysis Center for Software (DACS)

Affordable tools for improving reliability...Q.

This is a paid advertisement.
The DACS Journal 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. In 2009 DACS adopted 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 for First Publication Rights from the author to DACS for a period of ninety days following publication. After this ninety day period full copyright ownership returns to the author.

Although the DACS Journal 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 Journal web site (see https://journal.thedacs.com/) for further details.

To submit material (or ask questions) contact news-editor@thedacs.com

Recent themes include:

- Earned Value
- Software Testing
- Project Management
- Model Driven Development
- Software Quality and Reliability
- Cyber Security
ABOUT THE JOURNAL OF SOFTWARE TECHNOLOGY

DACS JOURNAL
EDITORIAL BOARD
John Dingman
Managing Editor
Editorial Board Chairman
Quanterion Solutions, DACS

Thomas McGibbon
DACS Director
Quanterion Solutions, DACS

Shelley Howard
Graphic Designer
Quanterion Solutions, DACS

Paul Engelhart
DACS COR
Air Force Research Lab

Morton A. Hirschberg
Army Research Lab (retired)

Dr. Kenneth E. Nidiffer
Software Engineering Institute

Dr. David A. Wheeler
Institute for Defense Analyses

Dennis Goldenson
Software Engineering Institute

John Scott
RadiantBlue Technologies

Dr. Richard Turner
Stevens Institute of Technology

ABOUT THIS PUBLICATION

The DACS Journal 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@thedacs.com

ADVERTISEMENTS

The DACS Journal 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@thedacs.com

COVER DESIGN

Shelley Howard
Graphic Designer
Quanterion Solutions, DACS

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 DACS Journal of Software Technology, Vol.15, No.1 February 2012. “

Requests for copies of the referenced journal may be submitted to the following address:

Data & Analysis Center for Software
100 Seymour Road
Utica, NY 13502-1348

Phone: 800-214-7921
Fax: 315-732-3261
E-mail: news-editor@thedacs.com

An archive of past newsletters is available at https://journal.thedacs.com.

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

ABOUT THE JOURNAL OF SOFTWARE TECHNOLOGY

The DACS Journal of Software Technology 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 Quanterion Solutions Incorporated.

Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the DACS. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the DACS, and shall not be used for advertising or product endorsement purposes.
Journal of Software Technology 15-1 February 2012: Cost Estimation and Systems Acquisition

IN THIS ISSUE

**Tech Views**

**Data as a Critical Service in Acquisition**
By Thomas McGibbon, DACS Director ................................................................. 3

**The Need for “Acquisition Visibility”**
By Mark Krzysko and Brian Baney ................................................................. 4

**DACS Software & Systems Cost and Performance Analysis Toolkit**
By Qi Li, Jo Ann Lane, and Thomas McGibbon ........................................ 12

**History is the Key to Estimation Success**
By Kate Armel, QSM ................................................................. 16

**An Innovative Approach to Quantifying Uncertainty in Early Lifecycle Cost Estimation**
By SEI Cost Estimation Research Group: Robert Ferguson, Dennis Goldenson, James McCurley, Robert Stoddard, and David Zubrow ................................................................. 24

**Parametric Modeling to Support System Acquisition**
By William Roetzheim ................................................................. 32

**Selling Your Software Estimate**
By Arlene Minkiewicz ................................................................. 40

**Outcome-Based Acquisition**
By Roger Stewart and Lew Priven ................................................................. 44

**A Roadmap for Reforming the DoD’s Acquisition of Information Technology**
By John M. Gilligan ................................................................. 52