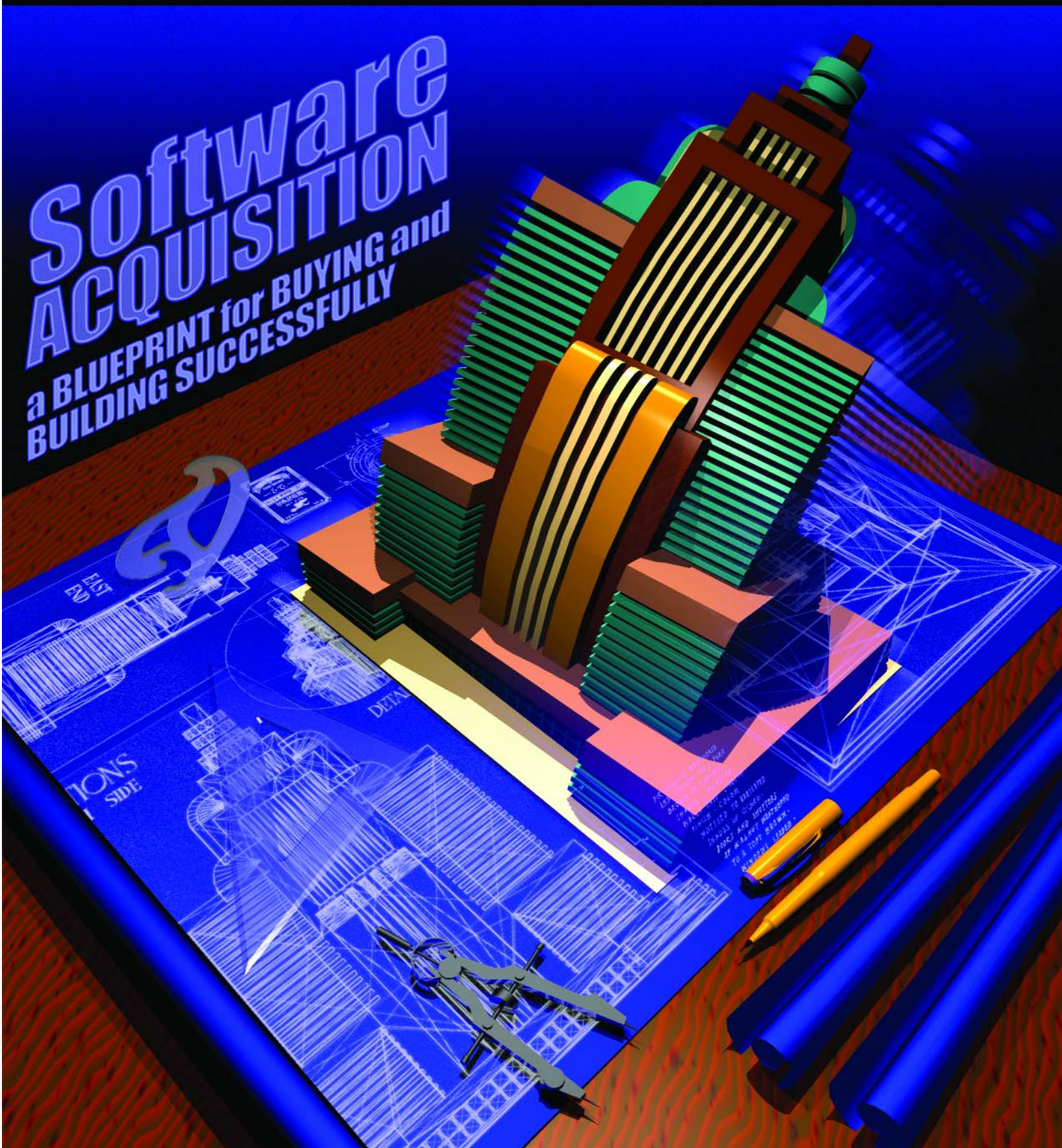
CROSSTALK

August 2002 The Journal of Defense Software Engineering Vol. 15 No. 8

Software Acquisition

a BLUEPRINT for BUYING and BUILDING SUCCESSFULLY



ON THE COVER
Cover Design by Kent Bingham.

CROSSTALK

SPONSOR	*Lt. Col. Glenn A. Palmer*
PUBLISHER	*Tracy Stauder*
ASSOCIATE PUBLISHER	*Elizabeth Starrett*
MANAGING EDITOR	*Pamela Bowers*
ASSOCIATE EDITOR	*Chelene Fortier*
ARTICLE COORDINATOR	*Nicole Kentta*
CREATIVE SERVICES COORDINATOR	*Janna Kay Jensen*
PHONE	(801) 586-0095
FAX	(801) 777-8069
E-MAIL	crosstalk.staff@hill.af.mil
CROSSTALK ONLINE	www.stsc.hill.af.mil/crosstalk
CRSIP ONLINE	www.crsip.hill.af.mil

Subscriptions: Send correspondence concerning subscriptions and changes of address to the following address. You may e-mail us or use the form on p. 25.

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

Article Submissions: We welcome articles of interest to the defense software community. Articles must be approved by the CROSSTALK editorial board prior to publication. Please follow the Author Guidelines, available at www.stsc.hill.af.mil/crosstalk/xtlkguid.pdf. CROSSTALK does not pay for submissions. Articles published in CROSSTALK remain the property of the authors and may be submitted to other publications.

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

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

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

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

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

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

Acquiring the Quality Products You Deserve

Software acquisition has been the theme of many issues of this journal because it is such an important topic in today's defense software community. With the continuing trend of outsourcing software development to industry, government acquisition professionals must continue to sharpen their acquisition skills to ensure they are purchasing quality software within budget and on schedule.

Some acquisition-related training courses often make the analogy of building a house to acquiring large software intensive systems. I can relate to this as several years ago, my husband and I acquired an acre of land and began plans to build our dream home upon it. Similar to an acquisition professional, we soon found ourselves knee-deep with bids from contractors. How did we know who to trust to carry out our dream? We were new to the area and had not had the luxury of watching homes be built over many years in the surrounding neighborhoods. The extremely costly bids promised superb quality and the cheaper bids promised the most for our money. This decision was not an easy one for a newlywed couple, especially for a couple of engineers. Perhaps you can imagine the research and analysis that we performed throughout this undertaking.

We did what we could to evaluate the past performance of the contractors as we toured the homes they had built and talked with homeowners. But all the decisions we made – from choosing the architect, to choosing the general contractor, to choosing a mortgage company – were not easy. We wanted the most for our money and felt we deserved good service and a high quality home in return.

As construction began on our home, my husband and I would regularly take an evening drive to look firsthand at the contractor's work for the day. I can still remember the evening we discovered a wall in our great room that was not supposed to be there. The framer had assumed we would want a wall between the breakfast room and family room. We were happy that we had caught this in time, but frustrated that we were the ones reading the plans for the framer and general contractor and reminding them of our requirements.

I am happy to say that we ended up with a beautiful home that we continue to be pleased with. And, my marriage did survive the "building your dream home" test. But I truly feel that as the customer and end user of this product, we would not have gotten the quality that we deserved if we hadn't verified our requirements by performing routine inspections.

As our theme articles discuss this month, there are many skills besides contractor selection and requirements verification that a software acquirer must be skilled at performing. We begin with *Applying the Software Acquisition Capability Maturity Model* by the Software Engineering Institute's Dr. Matthew J. Fisher, Wolfhart B. Goethert, and Dr. Lawrence G. Jones, which discusses a model framework that can help organizations improve software acquisition processes. *Achieving SA-CMM Level 2 at PM Abrams* by Col. Donald P. Kotchman, et.al., shows how an organization successfully institutionalized the model for its software acquisition processes and applied it to their M1A2 Abrams Main Battle Tank System Enhancement Package project.

Next is a re-notification of an April memorandum from the Under Secretary of Defense E.C. Aldridge Jr. on the subject of evolutionary acquisition and spiral development. The memo defines these approaches and clarifies how they can be used in providing "the warfighter with an initial capability, which may be less than the full requirement as a trade-off for earlier delivery, agility, affordability, and risk reduction."

Also, don't miss reading Lloyd K. Mosemann II's *Did We Lose Our Religion?*, a well-received speech at April's Software Technology Conference 2002 in Salt Lake City. Mosemann eloquently reminds us that government acquirers must be "smart enough to enunciate the basic processes that will be employed by the contractor" (to produce software that works on a predictable schedule and at a predictable cost).

A special thanks to our other authors who contributed such informative articles to this month's issue: Quentin W. Fleming, Lawrence H. Putnam, Ware Myers, David P. Quinn, and Lt. Col. Ken Alford, et. al. And finally, I wish those in the defense acquisition community the best of luck as they continue to implement processes and best practices to aid in obtaining the quality products you deserve.

Tracy L. Stauder

Tracy L. Stauder
Publisher

Applying the Software Acquisition Capability Maturity Model

Dr. Matthew J. Fisher, Wolfhart B. Goethert, and Dr. Lawrence G. Jones
Software Engineering Institute

Using poor management processes to acquire software can hold back the best software development processes. The Software Acquisition Capability Maturity Model® (SA-CMM®) was developed to help improve software acquisition processes so that organizations are better able to achieve acquisition goals. This article describes principles that helped successfully apply the SA-CMM and shares some lessons learned from successful and less successful SA-CMM-based improvement programs.

Government and industry are recognizing the need to improve the maturity of their software acquisition management processes. Unstable acquisition processes can impact a supplier's software development processes and result in substandard products. The Software Acquisition Capability Maturity Model® (SA-CMM®) was developed to help improve the acquirer's ability to manage such acquisitions by providing a mechanism to discipline the acquirer's software acquisition processes [1].

Organizations have used the SA-CMM not only to instill discipline in their software acquisition processes, but also in their general acquisition processes. This is possible because of the flexibility and adaptability of the SA-CMM for each organization's unique business needs. In this paper, we explore the features of the SA-CMM that support this flexibility and provide some lessons learned in applying the model to several different organizations' process improvement efforts.

What Is the SA-CMM?

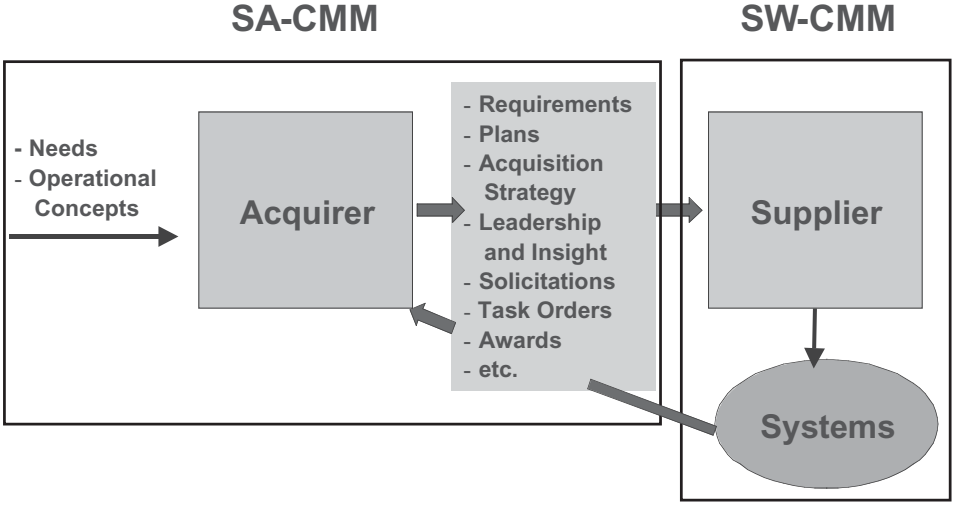
Capability maturity models are collections of features that reflect effective processes and practices for various disciplines. The first capability maturity model, the Software Capability Maturity Model® (SW-CMM®), was created by the SEI to help organizations improve their software development processes [2]. The SA-CMM was created to help organizations improve their acquisition processes.

There are other capability maturity models for personnel management and for systems engineering. The ongoing Capability Maturity Model Integration℠ (CMMI℠) effort is an attempt to combine several of these disciplines under one framework [3]. The CMMI model variant that partially includes the acquisition discipline has a product development perspective with acquisition seen as supporting the development.

The SA-CMM has been and is focused on software acquisition and management of the acquisition rather than development. Once the CMMI fully embraces the concepts and principles of the SA-CMM, it would be expected that the SA-CMM would be retired three years after. For now, the SA-CMM provides the comprehensive software acquisition focus.

The SA-CMM focuses on how the acquisition organization manages its internal business, not on how the supplier (developer) manages his development project. From another perspective, the SW-CMM describes the seller's (supplier's) role while the SA-CMM describes the buyer's (acquirer's) role in the acquisition process, as shown in Figure 1.

Figure 1: *Areas of Focus for Acquirer and Supplier Models*



If applied correctly, the SA-CMM results in an introspective view of an organization's ability to accomplish its software acquisition mission. Typically, such introspection reveals areas for improvement that include the following: the organization lacks institutionalized acquisition processes, the processes are inefficient and sometimes ineffective, organizational overlaps exist, responsibilities are not well defined, and visibility into projects is poor. Application of SA-CMM helps an organization understand the existence of these conditions. This realization provides the basis for developing improvement plans.

The SA-CMM architecture is structured into five levels of process maturity.[1] Maturity Levels 2 through 5 represent increasing organizational process maturity. Each maturity level (except Level 1) contains key process areas (KPAs), which are clusters of important, related practices. Table 1 depicts these levels and the key process areas at each level.

Note that the primary focus of Level 2 is basic project management within a single project. The Software Acquisition Planning and Solicitation KPAs provide the acquisition plans and requirements for the solicitation package and the resulting contract. Requirements for the solicitation and contract are developed under the Requirements Development and Management KPA. (These requirements are passed to the supplier through the solicitation package and contract.) Although the acquisition organization develops and manages the requirements from the beginning of the project, both the acquirer and the supplier have roles in managing requirement changes throughout the contract period of performance.

® Software Capability Maturity Model, SW-CMM, and Software Acquisition Capability Maturity Model, SA-CMM, are registered in the U.S. Patent and Trademark Office.
℠ CMM Integration and CMMI are service marks of Carnegie Mellon University.

While project management occurs throughout the acquisition, it is during the contract performance period that the Project Management KPA uses the Contract Tracking and Oversight KPA and the Evaluation KPA to collect information about supplier performance. Throughout the contract performance period, the Evaluation KPA focuses on objective evidence of product compliance with the contractual requirements. The Transition to Support KPA supplies life cycle support requirements to both the supplier and to the eventual support organization.

Level 3 focuses on process standardization and a more proactive approach among projects within the acquisition organization. For example, the Process Definition and Maintenance KPA focuses on creating and maintaining process definitions for the entire acquisition organization. The Project Performance Management takes several of the project-oriented activities from the Level 2 KPAs, primarily the Project Management KPA, and adds practices to foster better planning, communication, and cooperation. At the same time, the Contract Performance Management KPA adds more proactive and cooperative emphasis to the Contract Tracking and Oversight and Evaluation KPAs. Level 3 also supports its proactive management emphasis through the Acquisition Risk Management KPA. This includes risk management in the project management, planning, and contract management activities.

The SA-CMM Level 3 training program is established as an organization-wide training program. This concept supports the Level 3 requirement of *required* training for individuals, projects, and the acquisition organization in the Training Program KPA.

Level 4 emphasizes quantitative management of the processes and projects. Here, the application of quantitative data extends the Project Performance Management and Contract Performance Management KPAs from Level 3 to their respective Level 4 counterparts: Quantitative Process Management KPA and Quantitative Acquisition Management. In other words, Level 4 requires that project management and contract oversight be based upon quantitative data.

Level 5 emphasizes continuous and proactive improvement of the acquisition processes. The Continuous Process Improvement KPA uses Level 4 quantitative measurements to improve the acquisition processes while the Acquisition Innovation Management KPA includes technologies that support this process improvement.

These maturity levels provide a high-level road map for the continuous improvement of an organization's software acquisition process. As an acquisition organization attempts to achieve each higher level of maturity, its management must increase its involvement, leadership, and discipline. A maturity level is achieved by mastering all of the KPAs at that level, i.e., by achieving specified process goals within each KPA. The higher maturity levels of the model build on the lower levels. Indeed, some of the key process areas of the higher levels represent an improvement in the capability of the KPAs at lower levels as noted earlier.

Interpreting and Applying the SA-CMM

The following principles, most excerpted from the SA-CMM, provide some guidance on how to interpret the SA-CMM for specific acquisitions. In addition, these principles support the *flexibility* of the SA-CMM when applying it to specific organizations and acquisitions. However, interpreting the SA-CMM for a particular organization must be performed with professional judgment based upon both the knowledge of the principles below and experience in managing software acquisitions. Organizations that have adhered to these principles have typically achieved their process improvement goals more rapidly.

Interpret the Model in Light of Business or Mission Needs

The SA-CMM should be interpreted in the context of the business or mission needs of the organization. Effective and efficient acquisition processes are critical to successful process improvement, but the quality of their output can only be determined in the context of the organization's business needs. The SA-CMM should be tailored or adapted to fit the organization; the organization should not be restructured to reflect the SA-CMM.

Apply the Model in Terms of the Organizational Scope

The SA-CMM applies to the acquisition of all types of embedded and stand-alone software applications, including those where commercial-off-the-shelf (COTS) and non-developmental item software are being acquired, either as a part of a system or separately. Depending upon the mission of the organization, the SA-CMM may be used to improve acquisition processes for any type of product, not just software products.

One approach is to use the SA-CMM to identify and subsequently generalize acquisition processes for the products and services an organization wants to buy and then ensure the software aspects in the SA-CMM are included in these processes as appropriate. In this way, there is no need to tailor the SA-CMM. Inclusion of software under the general acquisition processes would satisfy the KPA goals of the model during an assessment. To illustrate this point, consider the Project Management KPA. The SA-CMM activities here focus on planning, staffing, and generally managing and controlling the acquisition project. These activities can be applied usefully regardless of the type of product or service being acquired.

The SA-CMM is designed to be sufficiently generic for use by any government or industry organization. When applying the SA-CMM to a particular organization, translations may be required (in addition to tailoring or adaptation of the model to fit a specific acquisition). Translation involves

Table 1: *SA-CMM Architecture*

Level	Focus	Key Process Areas	
5 Optimizing	*Continuous process improvement*	Acquisition Innovation Management (AIM) Continuous Process Improvement (CPI)	Higher Quality Productivity Lower Risk
4 Quantitative	*Quantitative management*	Quantitative Acquisition Management (QAM) Quantitative Process Management (QPM)	
3 Defined	*Process standardization*	Training Program (TP) Acquisition Risk Management (ARM) Contract Performance Management (CPM) Project Performance Management (PPM) Process Definition and Maintenance (PDM)	
2 Repeatable	*Basic project management*	Transition to Support (T2S) Evaluation (EVAL) Contract Tracking and Oversight (CTO) Project Management (PM) Requirements Development and Mgt. (RDM) Solicitation (SOL) Software Acquisition Planning (SAP)	
1 Initial	*Competent people and heroics*		Higher Risk Rework

mapping from the model's generic organization, language, and intent to how the acquisition organization does business. Also, the generic model terminology must be mapped onto the local situation; some examples are *contracting official*, *affected groups*, and *domain*.

The SA-CMM is not limited to formal contract acquisitions. It can be used to obtain software products from *in-house* groups. For this usage, the term *supplier* refers to the organization performing the required development effort. The term *project team* refers to the individuals within the acquiring organization who have an assigned acquisition responsibility, and the term *contract* refers to the agreement between the organizations.

Lessons Learned in Applying the SA-CMM

Because it is relatively new, the SA-CMM has not enjoyed the widespread awareness and acceptance as has the older SW-CMM. Thus, there is not a large amount of data to demonstrate proven return on investment. Evidence of a SA-CMM benefit is often anecdotal and typically relates to problems avoided, which are possibly more meaningful than return on investment for an acquirer.

Nevertheless, many organizations are applying the SA-CMM in their specific context because it makes good business sense to improve their acquisition processes. These applications vary considerably among organizations, ranging from awareness of the model, primarily through training, to full-scale process improvement, including assessments. Some organizations applying the model include U.S. Army, PM Abrams, Computer Sciences Corporation, the IRS Business Modernization Office, U.S Customs, the Department of Housing and Urban Development, and the General Accounting Office. All have had both good and bad experiences. We document here the following lessons that have been identified from applying the model. Many stem from not following the principles noted above or misinterpretations of the SA-CMM or process improvement in general.

Not a Silver Bullet for Troubled Projects

The SA-CMM is *not* a model for helping troubled projects. Some acquisition managers have heard of capability maturity models but do not have the requisite background to understand their intended uses. Instead, many of these managers are looking for the silver bullet that will retrieve their acquisitions projects from their death spiral.

The use of capability maturity models for process improvement is not a short-term effort. It is unrealistic to expect an immediate benefit of process improvement when a project is failing. Imposition of a process improvement effort may in fact have a negative effect on an already beleaguered acquisition. However, even in such projects, the SA-CMM may be employed as a diagnostic tool to understand where the project is going wrong. This would possibly allow a focused effort to correct the problems.

Prematurely Mandating a CMM Level Leads to Failure

Some acquisition managers have little knowledge of capability maturity models and their use in process improvement activities. Many have no problem imposing SW-CMM levels on their contractors. However, they have no knowledge about imposition of capability maturity model levels on their internal efforts, i.e., how long it takes, what it takes, needed cultural changes, and resources required. Instead, managers may draw a line in the sand and announce their organization will achieve SA-CMM Level 2 or 3 in six months. Artificial imposition of a capability maturity model timeline encourages organizational shortcuts in process documentation and implementation, thereby undermining the intended purpose of the capability maturity models.

"The SA-CMM has been and is focused on software acquisition and management of the acquisition rather than development."

Managers must understand the SA-CMM and what it takes in resources and time to achieve a certain maturity level. Understanding what their suppliers have gone through to achieve certain SW-CMM maturity levels might shed some light on what it takes. Data are certainly available. Also, realize that capability maturity models were developed with the original intent to improve processes, not achieve maturity levels.

Another indication of a similar misunderstanding is using the model only to achieve a maturity level rating, rather than instilling discipline into the process. This attitude may reflect a quest for status rather than a legitimate attempt to examine business needs and install process discipline to support these needs. The SA-CMM should be a means to an end, not an end itself.

Stabilize the Environment Before Attempting Process Improvement

Process improvement, whether using capability maturity models or not, is best done within the context of a stable environment. Some organizations do not understand their acquisition mission, do not have an organizational structure or skills to support this mission, or are evolving their acquisition processes as they learn how to accomplish mission requirements. For example, organizations that treat acquisition as simply managing the contractor significantly underestimate the challenges of program acquisition.

Before contemplating a process improvement effort, these areas of the environment should be sufficiently stabilized in order to facilitate success.

Treat Process Improvement as a Project

Process improvement experts agree – treat the process improvement effort as a well-run project. As a project, implement the following: develop reasonable plans with achievable goals based on business needs, do not try to improve everything at once, obtain long-term sponsorship and commitments, devote sufficient resources to the effort, and remember to plan deployment of process improvement efforts to the users.

Conduct Detailed Planning of the Acquisition

Some organizations believe that their acquisition projects do not have to be thoroughly planned. In other cases, there is an attitude that all planning resides with contractors, and the acquisition project simply follows the contractor's plans. It is clear that some acquisition projects do not understand what goes into a project plan, how to write a plan, and how to use a plan. There have been cases where project managers rely on planning templates without adapting the templates to their particular acquisition project. This results in devoting considerable resources to the development of plans that are not used. In such cases, organizational frustration may grow and resistance to process improvement may solidify.

The SA-CMM is based on the expectation that a mature organization and its

projects will do a thorough job of planning an acquisition. The resulting project planning documentation need not be any more extensive than that of any well-managed project.

Employ Demonstrated Expertise

In many cases, the acquisition organization has little knowledge or experience with SA-CMM and process improvement in general. This means it may have to contract externally for acquisition process improvement services.

If this is the acquisition organization's approach, it needs to obtain expertise in SA-CMM-based process improvement that can be verified and demonstrated. This expertise is especially critical in interpreting the SA-CMM in the context of the acquisition organization's environment and business paradigm.

Use the SA-CMM as a Starting Point

Acquisition organizations that buy software-intensive systems tend to believe the SA-CMM is not applicable to their acquisition processes since they buy systems, not software. One reason the SA-CMM was developed, as noted earlier, was to ensure these acquiring organizations realize the criticality of software in their acquisition.

However, organizations have used the SA-CMM as a foundation for process improvement in more general system acquisition processes, such as for systems and services. Such organizations have stabilized processes and have corrected or resolved long-standing control issues.

Moreover, if software is properly included within the context of these general acquisition processes and these processes include the practices or features of the SA-CMM, then assessments using the SA-CMM as a reference model can be successful. Of course, success depends on the rigor the organization uses when implementing its process improvements.

Summary

The SA-CMM was created for the discipline of software acquisition, i.e., to help organizations improve their software acquisition processes. The SA-CMM focuses on how the acquisition organization manages its internal business, not on how the supplier (contractor) does its business. The model was developed to be flexible enough in its application to be adaptive to a variety of organizations and their differing acquisition processes. Many of the principles documented in the model itself support this flexibility.

Currently, the SA-CMM is being used by organizations to improve acquisition processes. The exact usage ranges from simply learning about the model and its implications to extensive process improvement projects. We have found that some of these organizations have interpreted and applied the model incorrectly without following the principles discussed here. Such endeavors typically result in delays or cancellation of its process improvement efforts.

In general, we have found that the SA-CMM can be successfully applied to most acquisition organizations and their unique processes. When applied properly, the SA-CMM ensures that the acquisition organization is better poised to acquire the software product and services to meet the goals of the end users.◆

References

1. Cooper, J., et al. Software Acquisition Capability Maturity Model Version 1.02 (CMU/SEI-99-TR-002 ADA 362667). Pittsburgh, Penn.: Software Engineering Institute, Carnegie Mellon University, 1999. Available at <www.sei.cmu.edu/publications/documents/99.reports/99tr002/99tr002abstract.html>.
2. Paulk, M. et al., The Capability Maturity Model: Guidelines for Improving the Software Process. Addison-Wesley, Boston, Mass., 1995.
3. Software Engineering Institute. CMMI Product Suite. Pittsburgh, Penn.: Carnegie Mellon University, 2001. Available at <www.sei.cmu.edu/cmmi/products/products.html>.

Note

1. In CMMI terms, the SA-CMM is a staged model.

About the Authors

Matthew J. Fisher, Ph.D., is a visiting scientist in the Software Engineering Institute's (SEI) Software Engineering Process Management group. Prior to coming to the SEI, he worked for the U.S. Army as a civilian employee in several technology areas. During his tenure with the Army, he was deputy director of the Software Engineering Center for Army Intelligence Electronic Warfare systems. Fisher has a doctorate degree in electrical engineering from Drexel University in Philadelphia.

Software Engineering Institute
Carnegie Mellon University
4500 Fifth Avenue
Pittsburgh, PA 15213-3890
Phone: (412) 268-5877
E-mail: mjf@sei.cmu.edu

Wolfhart B. Goethert is a senior member of the Technical Staff in the Software Engineering Process Management program at the Software Engineering Institute (SEI). Prior to joining the SEI, Goethert retired from the U.S. Air Force and held numerous positions at the Illinois Institute of Technology Research Institute. He has authored and co-authored many technical reports on measurement and software cost estimation, and has been a presenter at many national and international software conferences on software measurement.

Software Engineering Institute
Carnegie Mellon University
4500 Fifth Avenue
Pittsburgh, PA 15213-3890
Phone: (412) 268-3889
E-mail: wbg@sei.cmu.edu

Lawrence G. Jones, Ph.D., is a senior member of the Technical Staff in the Product Line Systems Program at Carnegie Mellon University's Software Engineering Institute (SEI). Before joining the SEI, Dr. Jones served in the U.S. Air Force and is the former chair of the Computer Science Department at the Air Force Academy. Dr. Jones also serves on the Association for Computing Machinery Accreditation Committee and is the vice chair of the Computing Accreditation Commission.

Software Engineering Institute
1155 Kelly Johnson Blvd., Suite 111
Colorado Springs, CO 80920
Phone: (719) 548-4744
Fax: (719) 590-7652
E-mail: lgj@sei.cmu.edu

Achieving SA-CMM Level 2 at PM Abrams

Randy Schneider
Camber Corporation

Col. Donald P. Kotchman, David W. Schepke, Leonard M. Konwinski, and Edward Andres
U.S. Army

Mike Olsem
SAIC

In November 2001, the U.S. Army's Project Manager (PM) Abrams became the first federal organization to achieve Software Acquisition Capability Maturity Model® (SA-CMM®) Level 2 (Version 1.02) for its System Enhancement Package project. This article describes the path taken by PM Abrams to improve its overall acquisition processes, eventually leading to the achievement of SA-CMM Level 2.

The Project Manager (PM) Abrams M1A2 Systems Enhancement Package (SEP) Main Battle Tank is the world's premier ground combat platform. Developed to take advantage of the power afforded by integrated electronics and information technology, the PM Abrams M1A2 SEP tank provides mobile protected firepower to the digitized battlefield and rivals any U.S. Air Force fighter in terms of complexity and technology. Under-pinning this capability is an extensive network of interconnected software – more than 4 million lines of source code – driving eight major digital computer systems over a MIL-STD 1553 databus. The prime integration contractor must integrate various subsystems developed by multiple sources.

The PM Abrams M1A2 SEP project began in 1994 as a Pre-Planned Product Improvement (P3I)[1] program to enhance the capabilities of the fielded M1A2 tanks. Key features of this effort include fire control improvements, the addition of a thermal management system, incorporation of an under-armor auxiliary power unit, and the integration of Force XXI Battle Command Brigade and Below (FBCB2) command and control software.

The M1A2 SEP tank software includes subsystems provided by Horizontal Technology Integration (HTI)[2] program managers. The prime integration contractor, General Dynamics Land Systems Division, integrates these subsystems into the system software. The system level software is used by training device developers and by the off-vehicle diagnostics developers.

PM Abrams chose to assess their acquisition processes with the Software Engineering Institute's (SEI) Software Acquisition Capability Maturity Model® (SA-CMM®) in order to drive process improvement within the organization. There were several reasons for the decision to embark on this undertaking, but it was primarily due to these two factors:

- The need to level the playing field with our prime contractor, who was at the Capability Maturity Model® for software (SW-CMM®) Level 3.
- A recent personnel report indicated our command would likely lose more than half of its very experienced personnel before the year 2007. Thus, the SA-CMM process improvement effort was a means to capture this expertise before it was gone.

All levels of PM Abrams (management and staff) participated in preparing for the assessment. The program manager chartered a Software Acquisition Process Team (SAPT)[3] to coordinate the effort. On Nov. 19, 2001, PM Abrams became the first government acquisition organization assessed to be operating at Level 2 of the SA-CMM. This milestone capped off 18 months of effort that began with a baseline assessment in March 2000.

The M1A2 PM Abrams Main Battle Tank SEP was the project chosen for the assessment because it was the most comprehensive software acquisition project within PM Abrams. The processes used on the M1A2 SEP project closely reflect the processes used for all other PM Abrams projects.

Table 1: *SA-CMM Key Process Levels*

Level	Focus	Key Process Areas
5 Optimizing	*Continuous process improvement*	Acquisition Innovation Management (AIM) Continuous Process Improvement (CPI)
4 Quantitative	*Quantitative management*	Quantitative Acquisition Management (QAM) Quantitative Process Management (QPM)
3 Defined	*Process standardization*	Training Program (TP) Acquisition Risk Management (ARM) Contract Performance Management (CPM) Project Performance Management (PPM) Process Definition and Maintenance (PDM)
2 Repeatable	*Basic project management*	Transition to Support (T2S) Evaluation (EVAL) Contract Tracking and Oversight (CTO) Project Management (PM) Requirements Development and Mgt. (RDM) Solicitation (SOL) Software Acquisition Planning (SAP)
1 Initial	*Competent people and heroics*	

Higher Quality Productivity Lower Risk

Higher Risk Rework

Getting Organized

The SA-CMM model assesses five process levels. Table 1 describes these levels. The first order of business for the SAPT was to develop a charter and a plan of action to help us improve our processes and achieve Level 2 (as defined in Table 1). Having a well-defined scope of work is always a key element in any project's success. Putting extra effort, up front, to define this project's scope of work paid big dividends for PM Abrams. Working towards a common goal, using common terminology, and understanding the intent of the various Key Process Areas (KPAs) in the SA-CMM all contributed to PM Abrams' successful achievement of SA-CMM Level 2 for the SEP project.

Because PM Abrams buys systems and not just software, software vs. systems was one of our first scope of work considerations. We had to determine whether to use the existing SA-CMM or to tailor the model to accommodate our systems-oriented organization. We decided that the existing SA-CMM was flexible enough to accurately measure and assess how PM Abrams conducts its business relative to the KPA criteria.

Once these initial tasks were accomplished, we downloaded a survey from the SEI Web site intending to interview key

personnel and to determine how PM Abrams stacked up against the SA-CMM Level 2 criteria. It was here that we ran into a terminology problem. Some words used in the survey had different meanings to the personnel at PM Abrams than the SEI had intended. Thus, many of the original questions had to be re-written in terms that matched the intent of the question to the understanding of PM Abrams personnel. The resulting answers were then compiled and analyzed by the SEI to be used, eventually, by the baseline assessment team.

Baseline Assessment

We contracted with an outside organization to lead the SA-CMM baseline assessment. The resulting assessment team consisted of four internal PM Abrams members and four external members. We reserved two conference rooms, one to be used as the team's war room and the other to conduct the interviews. We then notified everybody well in advance as to who needed to be present for the interviews and when. We also discovered it is a good idea to periodically remind the same personnel regarding when they will be needed and to ask top management to reinforce the necessity of their availability.

We gathered the relevant documents together and catalogued them by creating an index on a CD. Here again, terminology created some problems as we asked for certain documents described by the model that existed at PM Abrams under different names.

We learned three key lessons as a result of our baseline assessment. First, the most significant lesson learned was mot to interview groups of people based on functional responsibility within PM Abrams. The functional groups interviewed could answer questions relevant to their division, but gave less informed and less accurate answers to questions outside their area of expertise. To correct the problem, we made sure that the Level 2 assessment group interviews included cross-functional expertise.

The second lesson learned was that we needed to come to a common interpretation of the Transition to Support (T2S) KPA within SA-CMM Level 2 as it applied to M1A2 SEP tank. To the credit of the SA-CMM model, enough flexibility existed to allow it to be adapted to the T2S life-cycle stage of the M1A2 SEP tank. The importance of understanding terminology and mapping it between the model and your organization cannot be over-emphasized.

In our third lesson learned, the baseline assessment indicated that PM Abrams was already performing most of the activities required by SA-CMM Level 2. We just were not documenting the procedures, policies, and charters we were following. PM Abrams, like most Level 1 organizations, relied on its experienced personnel to get the job done, which is fine as long as you never lose those experienced personnel. However, as indicated by the personnel report mentioned earlier, we knew we would lose the majority of these people during the next few years.

Institutionalization

Institutionalizing our processes was probably one of the most difficult aspects of process improvement to implement. The SEI defines institutionalization in the SA-CMM Version 1.02 as follows: "The building of infrastructure and corporate culture that supports methods, practices, and procedures so that they are the ongoing way of doing business, even after those who originally defined them are gone." Three elements played key roles in getting the desired institutionalization results we needed: training, developing user-friendly tools, and strong support from upper management.

"Three elements played key roles in getting the desired… results we needed: training, developing user-friendly tools, and strong support from upper management."

PM Abrams' personnel were trained extensively in both the SA-CMM and our policies, procedures, and the use of our process support tools. We felt that if people understood the maturity model, they would have an easier time understanding the relevance of all the questions being asked during the Level 2 interview sessions. This way they could answer the questions in the context of the model. Part of this training also included mapping terminology between PM Abrams and the SA-CMM.

Two key tools we developed as part of the process improvement effort were a process improvement Web site and the PM Abrams Digital Archive System (ADAS). These two tools were especially critical to institutionalizing our process improvement efforts. The PM Abrams process improvement Web site provided every employee with desktop access to all our plans, policies, standard operating procedures, definitions, process flow charts, and a monthly process improvement newsletter.

In conjunction, ADAS provided every PM Abrams' employee with desktop access to almost all of PM Abrams functional documents, both historical and current. ADAS is basically an Oracle database that allows the user to search by title, description, point of contact, division, or nomenclature and retrieve the actual document to their desktop (see ADAS sidebar on page 10 for additional discussion).

Strong management support from the very beginning was the third key to our success. Top management provided consistent oversight through the SAPT to ensure that PM Abrams' personnel were using these tools and getting the desired training. This, in turn, resulted in the level of institutionalization required to be assessed as a Level 2 project.

Gap Analysis and Mapping

Gap analysis is an invaluable process/tool that we recommend be done before a CMM assessment. It is a macro-level preassessment that identifies organizations' strengths/weaknesses and compares them to a given CMM level. However, it is extremely important that the organization is provided with this gap analysis information in plenty of time to take the necessary corrective action before the actual assessment.

A tool we developed internally to support the gap analysis process was a type of KPA mapping. Our Level 2 KPA mapping, as shown in Table 2 (see page 11), basically was a spreadsheet matrix that mapped each KPA goal, commitment, ability, activity, measurement, and verification to a specific process, policy, standard operating procedure (SOP), or document or artifact within PM Abrams. To satisfy the SA-CMM requirements for each KPA, at least one "X" must be present in each column. In addition to assuring we had sufficient coverage of each KPA for Level 2, this tool also saved a tremendous amount of time during the actual assessment by providing a cross-reference or road map from the KPA being assessed to the actual document or artifact that satisfied the

requirement. This was much more efficient than having to search independently for a document or artifact, since our documents were electronically hyperlinked to corresponding SA-CMM KPA criteria.

SOP Process

Obviously SOPs and policies must be in place for any SA-CMM assessment. The gap analysis helped us identify which SOPs and policies needed to be documented. We developed a very effective and efficient process for implementing our SOPs and policies for the SEP project at PM Abrams.

First we developed a standard template to follow for writing SOPs. This was posted on our SA-CMM Web site and made available to the entire Abram's work force. Next we ensured that an actual subject matter expert(s) was responsible for writing or updating the SOP or policy. Input for the SOPs was solicited from all relevant personnel to help create buy-in among the work force. Then the SOP was submitted to the SAPT for initial review. From there it was routed to the management steering group, and then ultimately to the deputy project manager for final approval.

Any reviewer could request an addition or change to the document, but the resulting modification would have to go through the entire review process again. For that reason we tried to consolidate as many changes as possible before sending it through the review cycle again. After final approval, the policies and SOPs were posted on our SA-CMM Web site.

This process resulted in policies and SOPs that were useful and relevant to the organization. This SOP effort also went a long way in helping PM Abrams document the expertise of our personnel – one of our principal goals in the process improvement effort.

PM Abrams Digital Archive System (ADAS)

The PM Abrams Digital Archive System (ADAS) began in May 2000 as an initiative to reduce paper copies of documents by imaging them to CD disks and cataloging them in a Microsoft Access Database. ADAS also played a key role in our successful achievement of SA-CMM Level 2 by allowing the assessment team to search and quickly retrieve documents as requested. Initially, PM Abrams successfully imaged more than 4,000 documents (representing over 500,000 pages), reducing the need for multiple file cabinets. After the imaging was completed, a user needing a specific document would fill out a request form and submit it to the ADAS administrator who would then forward an electronic copy to the requestor.

Later, a process was developed to improve turnaround time and user accessibility to these archived documents. The digitized documents were warehoused in an Oracle database and accessed by any authorized user via the local PM Abrams process improvement Web site. The ADAS user/database interface was developed using Oracle Forms. This Web-based, real-time document retrieval system now allows the user to both view and submit documents into ADAS.

Document security is managed by built-in Oracle Row-Level Security thus ensuring users only have access to appropriate data. To prevent any potential loss of information, the ADAS database is backed up regularly.

The following system requirements must be met on the user's PC in order for the program to operate:

- **Installed Oracle JInitiator Applet.**
- **Internet Explorer 5.0 or above.**
- **Windows 95/98/ME/NT/2000.**
- **Pentium 90Mhz or higher.**
- **16MB of system RAM or higher.**

The process is simple for any authorized user to follow. Once the applet is started, a valid username and password will take the user to a search screen. Here, several variables can be used to search for specific documents, including author, any word within the document's title, a short description of the document, nomenclature, part number, or the name of any of the divisions within PM Abrams. The system will return a list of all documents containing the input criteria. At this point the user can select the document he/she wishes to use.

ADAS is an excellent way to share information, archive historical documents, and ensure personnel are using the latest approved document. To date there are more than 5,600 stored documents in ADAS, available to three different access groups: public, government, and management. ADAS is predicted to grow to more than 10,000 documents in the next year and will continue to improve communications within PM Abrams. ADAS has improved the management of documents and increased awareness about specific program information. ADAS has improved interdisciplinary communications and taken the PM Abrams program one step closer to a paperless environment. ADAS will continue to be refined and revised as technology changes and as we continue to receive feedback.

Transition to Support

As discussed previously, the T2S KPA proved to be a major stumbling block during the original SA-CMM baseline assessment. Since PM Abrams has not (yet) transitioned the software (and its supporting hardware) to another maintenance organization, there was much discussion during the baseline assessment as to whether this KPA was relevant to PM Abrams. Unable to reach a consensus, the T2S KPA was not assessed during the baseline assessment.

During the months prior to the Level 2 assessment, the SEI and PM Abrams were able to demonstrate that the T2S KPA does allow for a transition to internal software support. (This was the interim reality at PM Abrams made necessary by the ongoing evolution of the tank software systems.) The T2S KPA also allows for the eventual transition to support by another organization prepared to accept this responsibility (the long-term goal for PM Abrams).

After the tank has been delivered to the acquiring organization, the capability to rapidly deploy software changes must be maintained throughout the tank's 30-year life cycle. A plan for the eventual transition of the tank system to a support agency is essential to ensure that system readiness is maintained. PM Abrams demonstrated to the SEI their readiness for this eventual transition to support. This section describes the M1A2 SEP program's approach to meet the SEI's SA-CMM Level 2 T2S KPA and ensure that an adequate life cycle software support management process was in effect for the M1A2 SEP.

SEP software changes were driven by evolving mission requirements, HTI mandates, systems upgrades, obsolescence, and field problem fixes. The complexity involved in orchestrating the implementation of so many external software inputs led to the decision to keep the SEP software acquisition management function within PM Abrams through the end of production. As such, the T2S plan took on a somewhat different flavor than traditionally expected, highlighting one of the primary benefits of using the SA-CMM model – adaptability. The model was flexible enough to allow it to be tailored to meet the unique needs of the PM Abrams program.

Program Management Key Process Area
Mapping Abrams Processes to
KPA Verifications (V), Measurements (M), Activities (AC), Abilities (A), Commitments (C), and Goals (G)

PM Abrams Division	Processes	Abrams SOP#	Artifact	V1	V2	M1	AC1	AC2	AC3	AC4	AC5	AC6	AC7	A1	A2	A3	A4		C1	C2		G1	G2
PMO	Program Manager & Project Managers have a charter.		PM Charter & APM Charter																x	x			
ALL	Each Division Manager has a "Roles and Responsibilities" document that defines each division's functions.		Briefing Charts	x																x		x	
PMO	Multi-year Authority		Multi-Year Contract						x			x	x		x				x	x		x	x
PMO/Engineering	A CPR is required from the contractor for every major program. Tracks project progress.	AB-006	Cost Performance Report (CPR)	x	x	x				x	x	x	x	x	x	x	x			x		x	x
ALL	Weekly PM report documenting indivduals key activities/accomplishments.	AB-003	One - liners	x	x	x											x					x	
PMO	O & S ownership cost reduction.	AB-007	TOCR Plan (PILOT)		x								x						x	x		x	
PMO	Cost Validation (AMSTA-RM-V)	AB-008	E-mail Certification from Cost Analysis Directorate			x		x														x	
ALL	PATs are formed for major projects.	AB-005	PAT Charter				x	x			x			x	x				x	x		x	x
PMO	Budget Execution	AB-043	Customer Files (MIPRs & customer checks) , TACOM Ledger (SOMARS), PM Ledger (PEST), Procurement Work Directive (MIPR), DASIS	x		x					x	x	x		x							x	x
PMO	Internal Controls	AB-044	Obligations Forecast (PROPS), FLASH Report, Variance Analysis, Year-end Certification Form, ULO Printout	x		x					x	x	x		x							x	x
PMO	Budgeting	AB-045	P-Forms & R-Forms (PB, POM, BES)	x	x	x									x							x	x
PMO	Report Management Activities to DA (monthly) regarding M1A2 status.	AB-009	PM-Monthly Report (MAPR)	x	x	x				x	x		x	x	x		x			x		x	
PMO/ALL	Document agreements between PM Abrams and other Gov't & non-Gov't entities.		MOA					x	x						x					x		x	
PMO	Ensure that policies, procedures, specifications, etc. are being followed.		Audit Reports	x	x	x	x	x		x	x	x		x	x				x	x		x	
PMO	Annual documentation of major events that occurred in the program.	AB-051	Annual Command History	x	x	x														x		x	
PMO	Historical documentation of PM Abrams events (updated annually).	AB-052	PM Abrams History	x	x	x														x		x	
PMO	Life Cycle Cost Estimates (done as needed).		POE	x	x					x		x	x		x		x		x			x	x
PMO	General Cost Estimates/ Economic Analysis (done as needed).		Price & Availability Reports, Various Reporting Formats	x		x				x	x		x									x	

Note: The first four column headings denote the folowing: the relevant organization (division) within PM Abrams, a brief description of the process currently in use, any applicable SOPs, and all relevant documents/artifacts. The processes and artifacts columns contain terms with acronyms that may be unique to the U.S. Army or PM Abrams. Such documents and processes should correspond to the unique way your organization currently conducts business.

Table 2: *Project Management KPA (Level 2) Mapped to Internal PM Abrams' Practices*

Transition Steps

From the model, the purpose of transition to support is to provide for the transition of the acquired software products to the software support organization. This effort should begin with initial program planning and the earliest definition of software requirements, and end when the responsibility for software acquisition shifts to the identified software support organization. From PM Abrams' perspective, transition to support requires that plans for transitioning software be developed and executable. Resources required for this transition are identified, funded, and available when needed. The software support contractor team and software support organization must be fully knowledgeable of software engineering and support environments. The development and configuration control infrastructure must be defined and maintained throughout the transition process.

The model requires that a transition to a software support organization must be planned. Since PM Abrams software development for the SEP program is still a work in progress with a projected development requirement for several years, the assessment focused on the transition from engineering and development to logistics and field support of the software products.

Even though acquisition management currently resides in the project office, transition to a support organization is well under way. The eventual software support organization has been identified. The Next Generation Software Laboratory in the Tank-Automotive Research Development and Engineering Center (TARDEC) has demonstrated the capacity and capability per the SA-CMM model to provide software acquisition and development support. This organization currently does software support for the standard M1A2 software. Personnel from TARDEC are dedicated to PM Abrams' software acquisition and currently work closely with their PM Abrams counterparts.

An M1A2 systems integration laboratory is up and operational and plans and budgets are in place for SEP. Support agreements are in place with TARDEC, General Dynamics Land Systems (GDLS, the tank's prime contractor), and PM Abrams to ensure continued support for the M1A2 SEP tank. From an acquisition perspective, the related development and support contracts include provisions to

ensure that a transition to support can take place. Ownership of data rights, support documentation, and system and software integration laboratory (SIL) components are retained by PM Abrams.

In preparing for the assessment and to improve the overall acquisition process, PM Abrams did ensure that the following activities must be performed while preparing for a potential transition to a new support agency:

- Essential engineering documentation must be identified, developed, reviewed, delivered, and maintained.
- The infrastructure and Computer Aided Software Engineering (CASE) tools used to develop, compile, build, and test the software products must be identified and procured by the supporting agency.
- The supporting agency must be equipped with the target hardware and the platform system for SIL.
- The supporting agency must be equipped with compatible development platforms in order to properly host the CASE tools needed for development.
- The supporting agency must have the facilities, expertise, and domain knowledge.
- Configuration items shall be defined and placed under configuration control.
- Source code files and program listings (i.e., software product specifications) must also be provided to the supporting agency.
- Funding for support planning, preparation activities, and the software support itself should be included in the life-cycle budget planning.
- Transfer of proprietary rights, licenses, and warranties to these software products have to be planned with future support and modifications considered.

Long-Term Support

The PM Abrams SEP tank, as with many major weapons systems, has a life-cycle span of 30 years or more. The software that brings these systems to life will evolve over its life span to meet the challenges of new threats, new operating environments, and hardware/software obsolescence. There is always a possibility that the original developer and/or the acquiring organization may eventually cease to support these software products. The original developing organization (i.e. contractor) could decide that supporting the software product may not be financially rewarding and thus not in their best business interests. The original developer may be out of business while the system still has many years of use left in its life cycle. The acquiring organization may be refocused, downsized, or eliminated. The people who originally developed and acquired the software products will most likely not be the people who support and manage the software.

In addition to the long-range plan to transition the software to a support organization, PM Abrams is planning to ensure that software support can continue should the original software developers or PM Abrams itself cease to exist.

The M1A2 SEP T2S strategy will ensure that all necessary engineering tools and practices are in place and updated to support a mission transfer. While often difficult to focus on T2S during the early phases of the acquisition process, failure to include postproduction acquisition planning with T2S in mind can leave your system unsupportable.

"The most significant lesson learned was not to interview groups of people based on functional responsibility within PM Abrams."

Summary

Several personnel issues were very important to our successful achievement of SA-CMM Level 2. First, our top management believed very strongly in the benefits of process improvement and the SA-CMM. They ensured that process improvement was a top priority in the organization and provided the resources necessary to accomplish this task. PM Abram's management steering group provided experienced people from each of their divisions to serve on the SAPT. The SAPT was then trained in process improvement and given the time necessary to accomplish these goals. Without the strong commitment from top management, we probably would not have been able to attain our goals in process improvement.

Terminology was another area in which extra effort early in our process paid big dividends. We discovered from our baseline assessment that PM Abrams personnel were having difficulties answering CMM questions because of some terminology differences. For example, *project manager* in the CMM terminology was equivalent to *product manager* in PM Abrams' terminology. So we identified numerous terminology differences and addressed them through work force classroom training, mock-assessments, and e-mails.

As a result of these efforts, we experienced little (if any) terminology difficulties during the actual SA-CMM Level 2 assessment.

Metrics/measurements turned out to be less of an effort than we originally imagined. The key for Level 2 was to ensure that we were tracking the internal status of our project for each KPA. It can be as simple as meeting minutes or tracking the status of assigned tasks. Even though we had some more formal measurement tools in place, internal status tracking is what is required for the SA-CMM Level 2 Measurement & Analysis category.

As for PM Abrams' future plans regarding SA-CMM, initially we intend to expand our Level 2 processes to all projects within the PM. Beyond this, PM Abrams will focus on improving our risk management activities – a key SA-CMM Level 3 KPA.◆

References

1. Cooper, J. et al. Software Acquisition Capability Maturity Model. Version 1.02 (CMU/SEI-99-TR-002 ADA 362667). Pittsburgh, Penn.: Software Engineering Institute, Carnegie Mellon University, 1999.
2. United States Department of Defense, Defense Systems Management College, Acquisition Policy Department. Glossary. Fort Belvoir, Va.: Defense Acquisition University Press. 10th Edition. Jan. 2001.

Notes

1. Pre-Planned Product Improvement (P3I) stands for planned future improvement of developmental systems that go beyond the current performance envelope to achieve a needed operational capability.
2. Horizontal Technology Integration (HTI) stands for the integration and application of common technologies across multiple systems.
3. The Software Acquisition Process Team (SAPT) is PM Abrams equivalent to the Software Engineering Process Group (SEPG).

About the Authors

Col. Donald P. Kotchman has served as commanding officer, A Company, 801st Maintenance Battalion, 101st Airborne Division (Air Assault); executive officer at Watervliet Arsenal and subsequently at the U.S. Army Tank-Automotive and Armaments Command; acquisition management officer, Army Material Command; and support operations officer for the 702d Maintenance Battalion, 2d Infantry Division, Camp Casey, Korea. Col. Kotchman has a bachelor's of science degree in applied science and engineering from the U.S. Military Academy, a master's of science degree in mechanical engineering from Rennselaer Polytechnic Institute, and a master's of science degree in national resource strategy from the Industrial College of the Armed Forces.

Abrams Tank System
SFAE-GCS-W-AB
6501 E. 11 Mile
Warren, MI 48397-5000
Phone: (586) 574-6885
E-mail: kotchmad@tacom.army.mil

David W. Schepke is a GS-13, operations research analyst for PM Abrams Tank Systems. He has more than 11 years of experience with the U.S. Army Tank-Automotive and Armaments Command and has spent the last two years focusing on process improvement efforts at Abrams Tank Systems. He has a bachelor's of science degree in marketing research from Oakland University, Rochester, Mich. and a master's degree in business administration from Wayne State University, Detroit, Mich. Schepke graduated from the Department of Defense's yearlong Executive Leadership Course.

Abrams Tank System
SFAE-GCS-AB-P
6501 E. 11 Mile
Warren, MI 48397-5000
Phone: (586) 753-2333
Fax: (586) 574-6237
E-mail: schepked@tacom.army.mil

Leonard M. Konwinski is the chief of the Logistics Readiness Branch at PM Abrams in Warren, Mich. He manages PM Abrams' logistics support requirements to include on-and off-vehicle diagnostics development and fielding, and software logistics for the PM Abrams family of vehicles. Konwinski has a bachelor's degree in journalism and a master's degree in public administration from Central Michigan University.

U.S. Army TACOM
SFAE-GCS-AB-LR
Building 229 M.S. 506
6501 E. 11 Mile Road
Warren, MI 48397-5000
Phone: (586) 574-8206
Fax: (586) 574-8208
E-mail: konwinsl@tacom.army.mil

Edward Andres is a project engineer for the Program Manager's Office for PM Abrams Tank Systems. Andres leads an Integrated Product Team that is responsible for managing the M1A2 System Enhancement Package software programs. He is a member of the Software Acquisition Process Group responsible for software acquisition process improvement. Andres has a bachelor's degree in electrical engineering from Wayne State University, Detroit, and a master's degree in software engineering from Oakland University, Rochester, N.Y.

U.S. Army TACOM
SFAE-GCS-AB-SI
Building 229 M.S. 506
6501 E. 11 Mile Road
Warren, MI 48397-5000
Phone: (586) 753-2386
Fax: (586) 574-8522
E-mail: andrese@tacom.army.mil

Mike Olsem is a senior software engineer for Science Applications International Corporation with more than 24 years of related information technology experience. He is currently working on the Capability Maturity Model® for Software process improvement effort project for PM Abrams. He has been a key factor in software engineering, developing a database search and retrieval system, as well as process improvement at PM Abrams. Before supporting the U.S. Army, Olsem supported the U.S. Air Force in the Software Technology Support Center. He has a master's degree in computer science from Brigham Young University.

Abrams Tank System
SFAE-GCS-W-AB-S
6501 E. 11 Mile
Warren, MI 48397-5000
Phone: (586) 574-6760
E-mail: olsemm@tacom.army.mil

Randy Schneider works for Camber Corporation as a systems administrator for the PM Abrams Digital Archive System where he is responsible for the development, deployment, and fielding of the system. Schneider also supports the Abrams Tank System by serving as a member of the Software Acquisition Project Team. He holds certifications as a Microsoft Certified Systems Engineer, Novell Systems Administrator, and CompTIA A+. As a result of the expertise provided by his software certifications, Schneider is also supporting the development of the Abrams Integrated Data Environment solutions.

Camber Corporation
31201 Chicago Road, Suite B-202
Warren, MI 48088
Phone: (586) 753-2364
E-mail: rschneid@camber.com

Evolutionary Acquisition and Spiral Development

To clear up confusion about evolutionary acquisition strategies and the spiral development process, the under secretary of defense issued a memorandum in April 2002 defining these processes. That information is repeated in this article.

The publication of Department of Defense (DoD) Directive 5000.1 and DoD 5000.2 established a preference for the use of evolutionary acquisition strategies relying on a spiral development process. Since those directives have been published, there has been some confusion about what these terms mean, and how spiral development impacts various processes such as contracting and requirements generation that interface with an evolutionary acquisition strategy. The purpose of this article is to address these questions.

Evolutionary acquisition and spiral development are methods that will allow us to reduce our cycle time and speed the delivery of advanced capability to our warfighters (Figure 1). These approaches are designed to develop and field demonstrated technologies for both hardware and software in manageable pieces. Evolutionary acquisition and spiral development also allow insertion of new technologies and capabilities over time.

These approaches provide the best means of getting advanced technologies to the warfighter quickly while providing for continual improvements in capability. Evolutionary acquisition and spiral development are similar to pre-planned product improvement but are focused on providing the warfighter with an initial capability that may be less than the full requirement as a trade-off for earlier delivery, agility, affordability, and risk reduction.

Evolutionary Acquisition

Evolutionary acquisition is an acquisition strategy that defines, develops, produces or acquires, and fields an initial hardware of software increment (or block) of operational capability. It is based on technologies demonstrated in relevant environments, time-phased requirements, and demonstrated manufacturing or software deployment capabilities.

These capabilities can be provided in a shorter period of time, followed by subsequent increments of capability over time that accommodate improved technology and allow for full and adaptable systems over time. Each increment will meet a militarily useful capability specified by the user (i.e., at least the thresholds set by the user for that increment); however, the first increment may represent only 60 percent to 80 percent of the desired final capability.

There are two basic approaches to evolutionary acquisition. In one approach, the ultimate functionality can be defined at the beginning of the program, with the content of each deployable increment determined by the maturation of key technologies.

In the second approach, the ultimate functionality cannot be defined at the beginning of the program, and each increment of capability is defined by the maturation of the technologies matched with the evolving needs of the user.

Spiral Development

The spiral development method is an iterative process for developing a defined set of capabilities within one increment. This process provides the opportunity for interaction between the user, tester, and developer. In this process, the requirements are refined through experimentation and risk management, there is continuous feedback, and the user is provided the best possible capability within the increment. Each increment may include a number of spirals. Spiral development implements evolutionary acquisition.

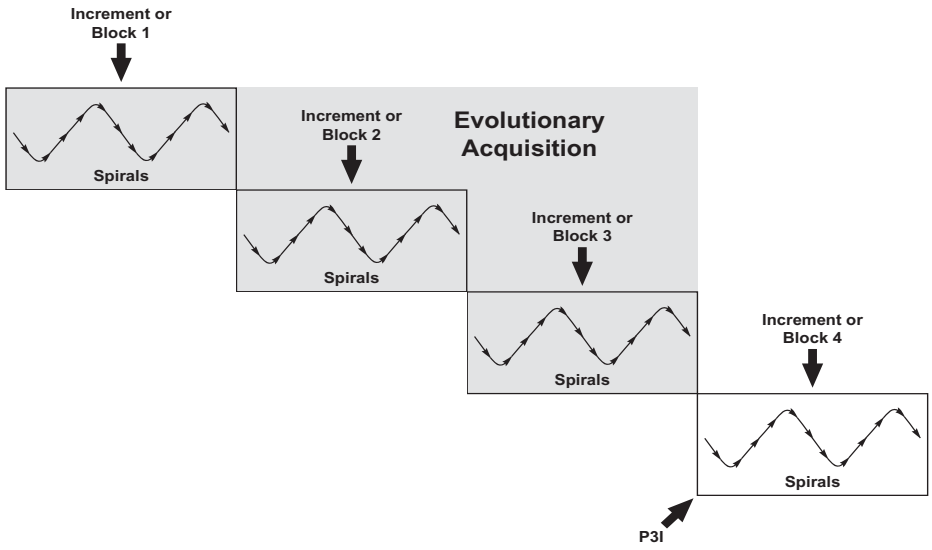
Increment or Block

An increment or block is a militarily useful and supportable operational capability that can be effectively developed, produced or acquired, deployed, and sustained. Each increment of capability will have its own set of thresholds and objectives set by the user.

Pre-Planned Product Improvement

Pre-planned product improvement (P3I) is a traditional acquisition strategy that provides for adding improved capability to a mature system.◆

Figure 1: *Evolutionary Acquisition, Spiral Development, and P3I*



Additional Information

For additional information or clarification, contact one of the following people:

Skip Hawthorne
Acquisition Initiative Office
Phone: (703) 697-6399
E-mail: skip.hawthorne@osd.mil

Ramona Lush
Acquisition Resources and Analysis Office
Phone: (703) 695-5166
E-mail: ramona.lush@osd.mil

Teaming Agreements Are a Lot Like Arranged Marriages

Quentin W. Fleming
Management Consultant

The outsourcing of information services, including the maintenance and updating of their related legacy software programs is becoming commonplace in industry. Sometimes such relationships are created using clearly defined roles and relationships. Other times, such services are simply scattered with overlapping or shared relationships. This author suggests using teaming agreements to precisely specify who does what and who is responsible for what.

Teaming agreements between corporations are a lot like *arranged marriages* within certain cultures of the world. The parents – mostly the fathers – get together and decide that one's son will marry the other's daughter. Period. End of discussion. The parents then meet – mostly the fathers – and introduce the young participants, who have no say in the matter.

Likewise, with many corporate teaming arrangements, one executive will meet with another executive and they will decide that their firms will join together on a new project. Period. End of discussion. The executives then meet to introduce the project participants, who have no say in the matter.

The funny thing is that arranged marriages between previous strangers most often work. Even funnier, perhaps, such arrangements between corporations and their projects also seem to work. Perhaps we in industry have learned something from the ancient ways.

Now that the U.S. Department of Justice and the European Community are starting to vigorously object to permanent consolidations (mergers) between international companies, we are starting to hear more about the formation of strategic teaming arrangements between what are otherwise competing firms. Hardly a week goes by that we do not read about major competitors forming some type of an arrangement or alliance, a strategic relationship, to go after a certain new project. And strangely, such arrangements seem to work.

We see this phenomena happening in all industries, but perhaps most particularly in the information technology (IT) outsourcing industry where new multi-year contracts are awarded almost monthly. These huge mega-deals are often beyond the capability of any single firm to support. But two or three companies acting in unison with each other seem to work nicely. One example is Electronic Data Systems Corp., who was awarded a far-reaching contract valued at as much as $6.9 billion over eight years to revamp the U.S. Navy and Marine Corps' computer system. Its key partners in the contract included WorldCom Inc. and Raytheon Company [1].

What is the fascination with teaming arrangements? Why are so many being formed? When one firm commits to joining forces with another firm, what does that really mean? What is the best approach for firms to take?

"We see this phenomena [teaming] happening in all industries, but perhaps most particularly in the information technology outsourcing industry…"

What are Teaming Arrangements?

As a starting point we should understand the concept itself. Teaming agreements in a nutshell are simply legal contracts between two or more companies. In teaming agreements, firms agree to do something, or to refrain from doing something. To be enforceable, such agreements need to be for a legal purpose.

Teaming agreements between one company and another means that two or more companies will join forces to go after a new segment of work, often a particular new project. Each company will commit something unique to the arrangement such as financial resources, key people, company assets, technology, etc., and each will expect to share in the risks and rewards of the endeavor. Perhaps a couple of specific formal definitions will help us to understand the concept.

Two leading authors in the field of project management have defined teaming arrangements in the following manner:

> An agreement of two or more firms to form a partnership or joint venture to act as a potential prime contractor; or an agreement by a potential prime contractor to act as a subcontractor under a specified acquisition program; or an agreement for a joint proposal resulting from a normal prime contractor-subcontractor, licensee-licensor, or leader-company relationship [2].

This may help. But perhaps another definition will better reinforce our understanding. Since many of us work on contracts funded by the U.S. Government, perhaps we should understand their perspective of such arrangements:

> An arrangement between two or more companies, either as a partnership or joint venture, to perform on a specific contract. The team itself may be designated to act as the prime contractor; or one of the team members may be designated to act as the prime contractor, and the other member(s) designated to act as subcontractors. When the characteristics of joint control (i.e., joint property, joint liability for losses and expenses, and joint participation in profits) are evident, then the teaming arrangement is a joint venture. When these characteristics are not present then the arrangement may more closely resemble that of a prime contractor/subcontractor [3].

There is one important point when

trying to understand the concept of strategic teaming: such agreements between two or more companies can be whatever these companies want them to be in their new relationship. Whenever two or more parties announce that they have formed a teaming alliance, the specific details of who is responsible for what are typically known only to the teaming participants, possibly their customer, as defined in their agreement. Teaming arrangements are the product of the parties involved:

> The strategic alliance is the parties' own creation. There are few laws constraining the teams to which the parties can agree … Parties to a strategic alliance agreement, therefore, need to be careful to state fully the terms of their alliance [4].

Good, bad or otherwise, a teaming arrangement between one company to another creates a unique arrangement. Great care must be taken to ensure that the strategic arrangement represents the intent of all parties.

Now let us get back to the matter of the outsourcing (IT) services. There are two common models commonly used to outsource these activities. Each paradigm places a different responsibility on the performing companies. These two approaches are 1) with use of a teaming agreement whereby we must buy our product from a certain company, or, we must sell our product to a certain company, or, we must split a project by some preset formula; or, 2) without a teaming agreement whereby we must cooperate with another company or companies to satisfy the ultimate wishes of the customer, no matter what.

To team or not to team, that is the question.

Model No. 1: Outsourcing IT Services with a Teaming Agreement

This first model employs a teaming agreement. Such arrangements specify precisely the roles of each party. The participants have an arrangement clearly delineating who is responsible for what. They have formed a precise relationship between the two or more participating companies: superior/subordinate, equal partners, 60/40 split, etc. By the terms of their teaming agreement, each company knows what is expected of it. Responsibility and authority will be clearly outlined.

This approach is illustrated in Figure 1, using as an example a recent United States Air Force (USAF) contract for the Aerospace Center support work at Arnold Air Force Base in Tennessee. In this case, the arrangement calls for a superior/subordinate relationship. This model is clear, clean, and workable. All parties, including the USAF buying customer and the two subcontractors, know who is responsible for the project: the Computer Sciences Corporation (CSC).

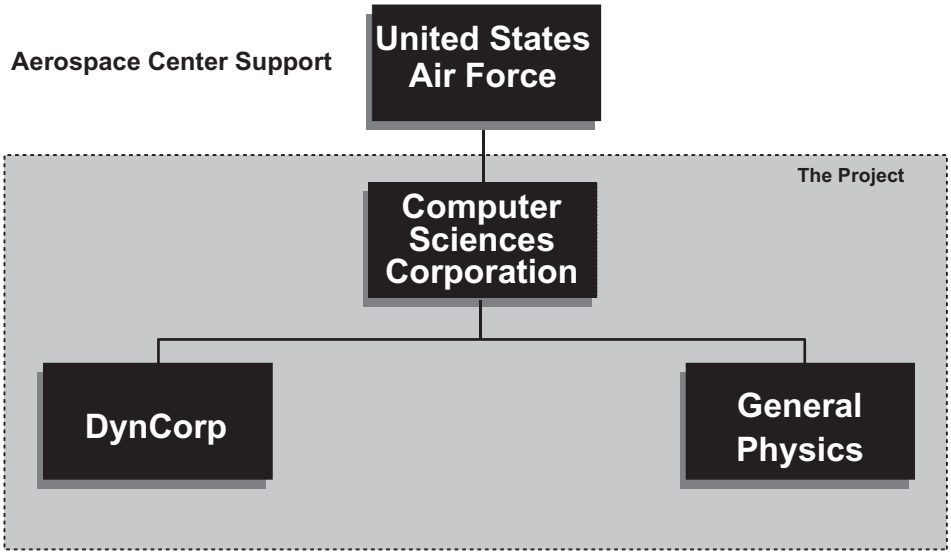
This model requires that the CSC buy certain previously defined scope-of-work from its two major team members for the duration of the agreement period, in this case three years. Typically under such arrangements, competition will be perpetually waived, and the principles must continue to buy (or sell) from the same source until the performance period is ended.

However, some teaming agreements will allow for either a pricing update or a competition to be held at a given future point in time. In this model, everyone clearly knows who is the boss of the effort. When things go right or possibly wrong, the USAF buying customer knows exactly who to hold accountable. The USAF has a direct privity of contract with only one company, CSC. In turn, CSC has a direct contractual relationship with both DynCorp and General Physics.

It should be stated that any teaming arrangement could (sometimes) be abused because the people in the trenches from the subordinate companies know that the prime contractor has no choice but to buy from them. However, if there is evidence that the subordinate firms are taking advantage of their legal agreement and not cooperating fully by providing either a reasonable price or adequate services, the best recourse is to elevate the issue back to the executives who consummated the deal in the first place.

In most cases, the established rapport between the executives who originated the initial agreement will be more than sufficient to bring reason back into such relationships. Much like the fathers in the ancient cultures, the executives expect – they will demand – that the teaming agreement works.

Figure 1: *With Teaming: A Superior-Subordinate Relationship*



Model No. 2: Outsourcing IT Services Without Teaming Agreements

In the second model, there is no teaming arrangement. The buying company simply expects that the chosen companies will work together in a cooperative way, under the direction of the IT buyer. Sometimes it works. Other times it is questionable.

We will use as an illustration the outsourcing of an IT effort that was done by British Petroleum Exploration (BPX), which started in 1993. BPX planned to outsource all its IT operations in an attempt to reduce its operating cost.

After BPX conducted an initial survey of how other IT services had been outsourced, they decided against using a single source supplier under a long-term arrangement, as other firms had elected to do. Rather, BPX chose to engage multiple contractors and insist these companies work in concert to provide the IT services. The company sent out a Request for Information (RFI) packet to 100 potential candidates indicating their intent to offer multiple contracts for all its IT work. A total of 65 companies responded to the RFI.

After a series of face-to-face discussions, BPX reduced the viable candidates' list down to only 16 firms. Next, BPX

went for a shortened list of only six firms. Weeklong sessions were held with these six final companies, resulting in the receipt of five compliant proposals. From these five proposals, BPX selected three firms to perform all its IT services. Multiple contracts were then awarded to the three selected companies.

This approach is illustrated in Figure 2: There is one overall IT project with three separate contracts, requiring each contractor to work with the other two to provide *seamless* services to BPX:

> Rather than totally outsource to one supplier, BPX hired three suppliers under an umbrella contract, which obligated the suppliers to work together [5].

Possibly, there could have been no contractual agreement between these three companies, since European antitrust laws may have prevented such teaming alliances:

> Although European antitrust laws prevented the three suppliers from joining in a formal alliance to deliver services to us (we have a separate agreement with each company), the companies agreed to provide combined services to all our sites [6].

The intent of BPX was to let the three contracted companies work out their own detailed interfaces, and to minimize the BPX management responsibilities:

> They wanted them to work together as a consortium – to present a united interface to the company, and deal with any issues amongst themselves, thereby minimizing BPX involvement [7].

How did this BPX contracting approach work? It appeared to be adequate; the needed services were delivered, but not without certain problems:

> The contracts were drawn up in ways that did not encourage cooperation between vendors. This left BPX with a range of inter-contract problems arising from what was described as *the cracks* between vendors. BPX ended up with the considerable task of having to manage not only each individual subcontractor but also the relationship and interfaces between them [8].

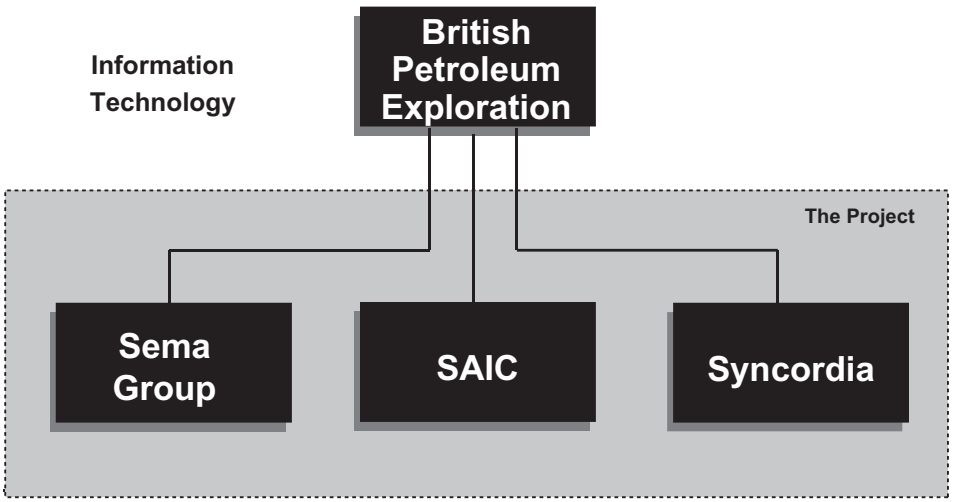


Figure 2: *Without Teaming: An Umbrella Contract for One Project*

At the end of their five-year contracts, all three companies were again retained by BPX, although in some cases their respective roles were diminished. But most significant perhaps, the vendor alliance concept was dropped at BPX. One of their managers later remarked:

> It's very difficult to get multi-vendors to work in alliance…We decided to go for the one-supplier option [9].

So much for the idea of getting cooperation from multiple suppliers.

Which Model Seems to Work Best?

In American football, there is a play called the *Hail Mary* pass. This pass is used whenever a team is in desperate straights, and they have no other course of action. The quarterback gets the ball, steps back, and throws a pass as far as he can in the direction of a cluster of players. Some of the players in the cluster are from his team, and some are from the other team. His silent prayer calls for someone on his team to somehow catch the ball. Sometimes it works. Most of the time it does not. It is a desperate measure.

There are two conditions calling for the use of the Hail Mary pass: 1) sheer desperation, and 2) no definitive plan of action. It would seem that the use of Model No. 2 as described above – the outsourcing of IT work without establishing clear lines of authority and responsibility – can be compared to the Hail Mary pass.

In the two models of IT outsourcing arrangements presented here, the first model calls for a teaming arrangement. The roles and relationships of all parties are established. There is someone specifically in charge, and all other participants are subordinate to that company. In the second model, the relationships between participants are not defined, and each entity is left to work out its role and relationship on its own. The question is, why would anyone choose to employ the second model?

Some will argue that the superior-subordinate model is unduly costly because the superior is given some value (a fee) for managing subordinates. This may be the case with the prime contractor getting a small (negotiable) fee for managing the subcontractors. But it would appear to be a value well spent. You always know exactly who is in charge, and who has the responsibility and authority for the project. You also know the total costs.

However, whenever you do not set clear lines of responsibility with your suppliers, someone has to manage the cracks and overlaps. Such management costs are often hidden, but they are real and are contained within the buyer's organization. When quantified, such supplier management costs will typically exceed the costs of a small management fee paid to a prime contractor to manage the entire effort. Not placing clear lines of responsibility with suppliers to save a small management fee is a false economy.

Others have suggested that by not specifically defining suppliers' roles with great precision, synergies between them will somehow emerge from the relationship, and each organization will excel with their respective contributions. This would seem to be an unduly optimistic approach.

Model No. 1, the use of a definitive teaming agreement, would appear to be most appropriate. In any business rela-

Coming Events

August 19-22
The 2nd Software Product Line Conference
San Diego, CA
www.sei.cmu.edu/SPLC2/

September 9-13
International Conference on Practical Software Quality Techniques (PSQT) 2002 North and International Conference on Practical Software Testing Techniques (PSTT) 2002 North
St. Paul, MN
www.psqtconference.com

September 24-27
Software Test Automation Fall Conference
Boston, MA
www.sqe.com/testautomation

November 4-8
Software Testing Analysis & Review Conference
Anaheim, CA
www.sqe.com/starwest

November 18-21
International Conference on Software Process Improvement
Washington, DC
www.software-process-institute.com

April 28-May 1, 2003
Software Technology Conference 2003
Salt Lake City, UT
www.stc-online.org

June 2-6, 2003
Software Management Conference and Applications of Software Measurement Conference
San Jose, CA
www.sqe.com/sm
www.sqe.com/asm

tionship, it is mandatory to know precisely who to praise when things go well, and who to hold accountable when things do not. There is nothing inherently wrong with teaming agreements as long as the buyer is aware of the teaming arrangement, and there is a competition held with other firms, or other teams.

Model No. 2 would appear to be fundamentally flawed, in the opinion of this author, as the BPX experience demonstrates in the following:

> Our outsourcing strategy has not always worked smoothly, we have encountered some bumps … While senior managers at BP and the three suppliers clearly understood the vision of seamless service captured in the framework agreements, their respective operations did not [10].

Perhaps we should again look to the ancient ways of arranged marriages between families. While in the Old World the families – mostly the fathers – would agree on the matchmaking. After the marriage, the participants decided who was responsible for what although there were certainly precedents to follow. What is being suggested is that companies should follow the ancient ways and let the parents – the corporate executives – decide what projects should be joined by other projects. However, such corporate relationships should not be left open to chance for the parties to work out.

In all cases, the executives who arrange the teams' formation should also insist on such agreements being reinforced in great detail. This includes defining who is responsible for what, and covering, among other things, the possibility of an early breakup, a dissolution of the arrangement, and a way to reasonably settle any disputed issues. In the modern world of marriage, we often refer to this arrangement as a *prenuptial agreement*.

Summary

In the opinion of this author, a combination of the Old World with the modern world makes the best form of a strategic teaming arrangement. The families (the corporate executives) should endorse any teaming agreement, but the details of their arrangement should be specifically spelled out: who does what, who is responsible for what, and how do we get out of this arrangement, all in the form of a prenuptial called a teaming agreement.◆

References

1. Prince, Marcelo, and Pat Maio. "EDS Wins Huge Contract To Revamp Military Computers." The Wall Street Journal. 6 Oct. 2000.
2. Cleland, Dr. David I., and Dr. Harold Kerzner. A Project Management Dictionary of Terms. New York, New York: Van Nostrand Reinhold Company, 1985: 253.
3. United States. Defense Contract Audit Agency. Contract Audit Manual. Part 7-1802, c, Jan. 1996.
4. Nibley, Esq., Stuart B., and Joseph J. Dyer. "Forming Strategic Alliances." Contract Management Magazine. Dec. 2001: 9.
5. Lacity, Dr. Mary C., and Dr. Leslie P. Wilcocks. Global Information Technology Outsourcing. Chichester, England: John Wiley & Sons, 2001: 24.
6. Cross, John. "IT Outsourcing: British Petroleum's Competitive Approach." Harvard Business Review May-June, 1995: 99.
7. Lacity, Dr. Mary C., and Dr. Leslie P. Wilcocks. Global Information Technology Outsourcing. Chichester, England: John Wiley & Sons, 2001: 225.
8. Ibid, p. 224.
9. Ibid, p. 231.
10. Cross, John. "IT Outsourcing: British Petroleum's Competitive Approach." Harvard Business Review. May-June, 1995: 100.

About the Author

Quentin W. Fleming is a management consultant and textbook author. His latest book, published by the Project Management Institute (PMI), is "Earned Value Project Management," co-authored by Joel M. Koppelman. Fleming developed two courses for the University of California at Irvine (UCI) for their Certificate Series in Project Management: "Earned Value Project Management," and "Project Procurement Management: Contracting, Subcontracting, and Teaming." He also serves on UCI's Project Management Advisory Board. Fleming was also on the core team that updated PMI's "A Guide to the Project Management Body of Knowledge," released in December 2000.

14001 Howland Way
Tustin, CA 92780
Phone/Fax: (714) 731-0304
E-mail: quentinf@msn.com

Control the Software Beast With Metrics-Based Management

Lawrence H. Putnam
Quantitative Software Management, Inc.

Ware Myers
Consultant

Living on a finite planet with a day length regulated by a sun implies that we have to complete the following: a quantity of work (measured by size), at some level of quality, within limits of time, and effort, at some degree of productivity. These are the five core metrics that enable managers to estimate and bid software projects, and control progress during the project. Higher executives and clients need to understand this pattern because, otherwise, they are the prime source of unrealistic expectations and the resulting failures.

Software development is a *beast* for the many organizations whose project failures keep baffling senior management and clients. But it is a beast that can be tamed. In fact, hundreds of organizations have already done so. Earlier disciplines show the way.

Executives control whole businesses with financial information collected and analyzed by accountants. Manufacturing executives cost factory production with data supplied by cost accountants. It follows then that software projects can be managed with appropriate information – progress indicators applied through metrics-based management.

Beyond the project, however, software development involves a broad range of stakeholders – client management, users, outsourcing organizations, and high-level management. As a result, not only must we establish measures and their collection and accessibility, but we also must communicate the nature of metrics-based management to these stakeholders.

Core Measures Are the Foundation

The measures underlying effective software management are effort, time, size, and a measure of quality or reliability such as the defect rate (or its reciprocal, mean time to defect). These key measures need to be clearly defined. The procedures for collecting them need to be clearly specified. The resulting data need to be kept in an accessible database.

Then, to use these four core measures for estimating, bidding, and project control, the industry has to establish the *relationship* between them. Let us back up a minute and consider the relationship behind any means of doing work:

Work *Product* (at a *Quality* level) = *Effort* over a *Time* interval at a *Productivity* level

One new term, productivity, has appeared. It appears that work measurement needs a fifth core measure, productivity. Where is this measurement to come from? Let us restate the foregoing relationship in software terms:

Size (at implied Quality) = Effort x Time x Productivity

From this expression, we see (rearranging it by algebraic methods) that Productivity is a function of size, effort, and time:

Productivity = Size / (Effort x Time)

In other words, software productivity comes from the core measures, and we can acquire them from completed projects. However, software productivity has been conventionally defined as size/effort (conventionally from economic theory, output/input). Hence, we need a new name for this version of software productivity that includes the *time* schedule. We called it *process productivity*. It is the productivity, not just of an individual programmer producing code (software lines of code/person-month), but of a project full of all kinds of software people operating over the time period of the project. Hence, this version of productivity includes the capability of requirements gatherers, analysts, designers, implementers, and testers. It covers the effectiveness of management, process, modeling, and tools. And not so incidentally, it is affected by the understanding with which stakeholders approach the software task.

Unfortunately (for the sake of progress in software estimating, control, and management) people have had some difficulty grasping the fact that the schedule planned at the beginning of a project does have an effect on the productivity that the software process can achieve.

To define this relationship more precisely, Putnam analyzed a broad database of completed projects. The general *work* relationship set forth above held, but the resulting equation turned out to be nonlinear, that is, effort and time were raised to powers. The basic software equation would then be:

Size (at some specified or attainable Quality)=Effort^a x Time^b x Process Productivity

The databases from which Putnam derived the relationship supply the values of the exponents, *a* and *b*, but those databases do not supply the value of process productivity at which the software organization will carry out its next project. This value is best calibrated from immediate past projects. Thus, process productivity becomes the *fifth* core measure, derived from three base measures.

The other input value to the estimating relationship – size – is appraised from what is known about the project at the time the estimate is prepared. If the size estimate must be made before much is known about the product, it will likely vary substantially from the eventual completed size. The corresponding effort and time estimates may, accordingly, be equally far from the eventual reality.

In software development, the relationship between effort and time is multiplicative. That is, if the size and process productivity terms are considered to be fixed in a particular application, estimators can shorten time only by increasing effort. Or, if they wish to reduce effort (and cost) they must allow more time. According to this relationship, effort and time must move in opposite directions.

Stakeholders Must Use the Relationship

Measurement systems may be operated in a detailed sense by a specialized measurement staff or even by project management. *Details* refers to the definition, collection, and databasing of the measures and the mechanics of using them for estimation and control. In addition, all the stakeholders from client management onward must understand metrics-based

management, in a broader sense.

For instance, it is often high-level management that sets the finish date to meet business commitments and lower levels that work out the requirements. Neither group may understand clearly that the amount of work needed to satisfy the requirements determines the effort. Moreover, they often fail to understand that the time (schedule) needed depends, as the foregoing relationship demonstrates, on the amount of effort financed. Furthermore, stakeholders have a strong hand in other factors entering into project estimating and bidding, such as the following:

1. A key measure in the estimating relationship is the size of the eventual product. Size is often measured in source lines of code, but it may be represented in any unit that represents the functionality to be produced by the project. Selection of that unit is a management chore, often involving the client as well.
2. The amount of functionality is clearly unknown when the eventual product is little more than a gleam in some high executive's eye. Estimation accuracy increases if it can be deferred until the product is delimited, defined, and de-risked. De-risked. Now there's a word you won't find in your dictionary. Books abound on software risk management. All we mean at this point is that you have to identify the important risks and mitigate them. That does not mean you have to solve them before bidding. It does mean you have to foresee an approach to solving them – an approach for which you can allow sufficient time and effort in the estimate and bid. Clients and management control how long making this firm estimate can be deferred, not the project or estimating staff.
3. Basically, there is a trade-off between effort and time. Within limits, planners can reduce time by increasing effort. They can reduce costs (effort) by allowing a little more time. They cannot have both at the same time, except by increasing process productivity and that takes time on a longer scale than project schedules. Therefore, stakeholders must be satisfied with a reasonable trade-off. There is no golden shortcut. That is a hard lesson for clients and higher management to recognize and accept.
4. Clients and management are up against business imperatives – get it done, in a short time, at an effort (cost) within the client's reach. But development is up against another set of imperatives symbolized by the relationship set forth above. The two sets of imperatives have to be reconciled. It helps if clients and executives have a grasp of the software relationship. It also helps if software managers understand business pressures. Sometimes they can strip down project functionality to come closer to business schedule-and-effort goals.

Finally, the software relationship is not a law of physics, that is, each term is uncertain to some degree. We may gain some insight by comparing the software process to a communication channel, as researcher Claude Shannon employed the concept [1]. Shannon originated the mathematical theory of transmitting information over a communication channel such as a signal over a wire.

1. The channel had a certain capacity or bandwidth – a transfer rate in bits per second.
2. It had a certain amount of noise, random electrical signals arising out of the environment that interfered with the transmission of the bits carrying the information.
3. As a result of capacity limitations and noise, some of the bits carrying information were distorted in transmission. That was an error rate.
4. This error rate may be reduced by improving the channel or adding error-correction algorithms.

Similarly, we may conceive of software development as taking place through a channel called a process, extending from systems definition and requirements capture to delivery.

1. This process has a certain capacity – mechanically measurable as bits at the input to the digital device that uses the software.
2. It generates a certain amount of noise, or defects, resulting from deficiencies in the process or errors by the people engaged in the process.
3. As a result of these deficiencies and errors, some of the output bit-stream is incorrect.
4. This defect rate may be reduced by improving the process (for example, instituting reviews) or correcting the product (for example, testing).

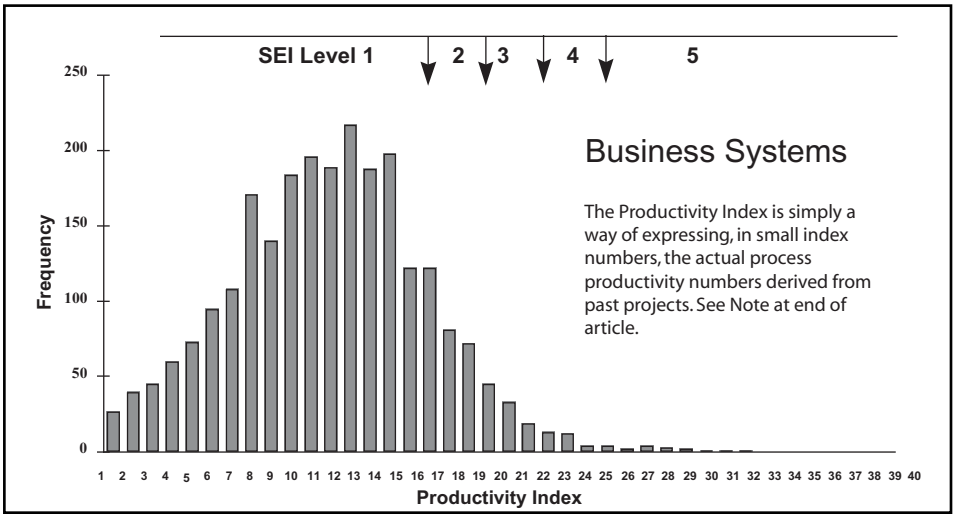
In this theory, the signal containing the information gets through the channel (usually – sometimes it is overcome by noise). However, the signal is normally afflicted with noise, and the receiver has to pick the signal out of the noise. In other words, a communication process is *uncertain*.

Similarly in software development, the product gets through the requirements-analysis-design-implementation-testing channel (maybe two thirds of the time), but the numbers that measure its progress through this channel are cloaked in uncertainty.

For example, the size is uncertain until the product is shipped. The process productivity is uncertain because the next project may have different staff, run into different problems, and so on. Consequently, the effort and time derived from the software relationship are also uncertain. Sometimes they are so uncertain that the project fails to complete. It is overcome by noise. It runs out of time and effort and has to be replanned.

By accepting this data uncertainty (or noise), planners can employ the principles of probability to provide enough time and effort to increase the chance of successful

Figure 1: *Quantitative Software Management – SEI Mapping: Example is for the process productivity of QSM's 5000-system database, distributed over a long range*



completion to whatever level management sets. For example, it could ask for 50 percent. At this point the odds are 50-50 that the project will complete within the expected effort and time estimates. It could set 80 percent, providing an increased assurance level by including time and effort buffers beyond the expected time and effort.

Process Productivity Proves its Worth

The Software Engineering Institute (SEI) at Carnegie Mellon University worked out the five-level Capability Maturity Model® (CMM®) in the late 1980s. Since then hundreds of software organizations have advanced from Level 1 to Levels 2 and 3. A few have made it to Levels 4 and 5. As a matter of observation and common sense, the effectiveness of these organizations has improved as they gained levels. Or, in the terms of the relationship set forth above [Productivity = Size / (Effort x Time)] their *process productivity* has gained.

Companies can compute process productivity from three of the core measures of completed projects. That makes it an objective number, not a matter of opinion. In measurement terms, it is an overall metric indicator of the effectiveness of software development of a project, a series of projects, or the development organization doing these projects. It is also a number that can be correlated with the SEI CMM levels and the work that Putnam did in the mid-90s, as brought up to date in Figure 1.

Note that the frequencies (vertical axis) conform roughly to the standard normal curve. Since many activities involving human measurements follow this curve, that fact suggests that the metric indicator, process productivity, is on the right track.

From here it is a simple mathematical step to express that increase in process productivity (with which not everyone is familiar) in terms of schedule, effort, and reliability improvement (which everyone understands). The result is presented in Table 1.

To manage software development intelligently, project managers need to understand the core measures and the mathematical relationship between them. They can then intelligently estimate, bid, monitor, control, and improve their ability to develop software over a period of time.

By basing the control of software development on these five core measures, management has the means to plan time and effort commensurate with the functionality (size) expected. By applying statistical methods to the acknowledged uncertainties of these measures, management can improve the odds of completing within a plan. By advancing through the CMM levels, management can greatly improve its core measures. The stakeholders need this understanding as well; it enables them to interact intelligently with the project managers and developers.

By extending an understanding of metrics-based management [2] to project managers, higher management, and stakeholders, the software industry can get the better of the beast.◆

SEI Level	Effort (Cost) Person-months	Peak Staff People	Schedule Months	Mean Time to Defect Days	Defects Remaining at delivery # of Defects
1. Initial	1542	91	26.0	0.34	255
2. Repeatable	831	60	21.2	0.52	137
3. Defined	241	26	14.0	1.18	40
4. Managed	92	14	10.2	1.78	21
5. Optimized	38	8	7.6	4.06	6

Table 1: *Increase in Process Productivity Expressed in Schedule, Effort, and Reliability. Example is for a 100,000-source-lines-of-code (SLOC) engineering system*

Further Information

More information on the five core measures [2] is available at: <www.qsm.com>.

References

1. Shannon, Claude E. "The Mathematical Theory of Communication," Bell System Technical Journal 27 (1948): 279-423, 623-656. Also University of Illinois Press, 1949.
2. Putman, Lawrence H., and Ware Myers. The Intelligence Behind Successful Software Management. New York, N.Y.: Dorset House Publishing, 2002 (to be published in the fall).

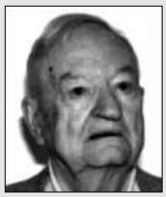
Note

The actual values are very large, ranging from 754 for Productivity Index 1 to 1,346,269 for Productivity Index 32, the highest shown on the figure. Each Process Productivity value is 1.27 times the previous one. The details depend upon the particular units of measurements used such as years for time and person-years for effort. We are trying to explain the general idea very briefly, not treat all the details.

About the Authors

Lawrence H. Putnam is president of Quantitative Software Management, which he founded in 1978. Previously, he spent 25 years on active duty, including tours in the Office of the Director of Management Information Systems, and the Assistant Secretary of the Army, Financial Management where he gained experience in software development from a top management perspective. Putnam is a graduate of the U.S. Military Academy at West Point.

Quantitative Software Management,Inc.
2000 Corporate Ridge, Suite 900
McLean,VA 22102
Phone: (703) 790-0055
Fax: (703) 749-3795
E-mail: larry_putnam_sr@qsm.com



Ware Myers is a contributing editor to *Computer*. Myers assisted Putman in 1981 with his first tutorial book for the Institute of Electrical and Electronics Engineers Computer Society. This was the beginning of a long writing collaboration. Myers is a graduate of Case Institute of Technology and has a master's degree from the University of Southern California.

1271 North College Avenue
Claremont, CA 91711
Phone: (909) 621-7082
Fax: (909) 948-8613
E-mail: myersware@cs.com

Did We Lose Our Religion?

Lloyd K. Mosemann II
Science Applications International Corporation

This article was presented as a keynote address at the Software Technology Conference 2002 in Salt Lake City in May and has been edited for length and clarity. The author takes the government to task for using religious-like belief systems rather than objective appraisals to build and buy software. The author says that best commercial practices do not exist in government contractors as they do in the real commercial world: in-house software expertise, a robust software development environment, and sound software architecture. While he is not suggesting that the government develops software in-house, the author does suggest that it needs enough in-house software expertise to know what it is buying.

In 1990, I declared that the 1980s were a lost decade from the perspective of software development progress. The question I posed was, "Will there be a breakthrough in the 1990's?" I went on to say, "It won't happen automatically; people are too satisfied with unsatisfactory ways. We dare not make the mistake of complacency a la the automobile industry; we must push awareness and resource commitment to get ahead of the power curve of demand."

In 1994, I closed the annual Software Technology Conference (STC) with the observation that the underlying need within the defense community is for *predictability*: "From a Pentagon perspective, it is not the fact that software costs are growing annually and consuming more and more of our defense dollars that worries us. Nor is it the fact that our weapons systems and commanders are becoming more and more reliant on software to perform their mission. Our inability to predict how much a software system will cost, when it will be operational, and whether or not it will satisfy user requirements is the major concern. What our senior managers and DoD (Department of Defense) leaders want most from us is to deliver on our promises. They want systems that are on time, within budget, that satisfy user requirements, and are reliable."

The question I pose now is: "Where are we today, and where will we be tomorrow?" Did we lose our religion?

Why did I use the metaphor of *religion*? Because religion is the traditional example of faith-based behavior – that is, behavior that is based on a belief system rather than on externally imposed rules such as the law of gravity or "she that has the gold, rules." Emotional discussions regarding whether Ada or C++ should be preferred are frequently described as *religious* arguments based on beliefs rather than facts.

Sadly, I still see the world of software being governed by religious-like belief systems rather than objective appraisals. When I left the Pentagon six years ago, I described some of what was happening as *bumper sticker* management, and the situation has not changed for the better. I sometimes have the feeling that the blind are leading the blind – the leadership is blissfully ignorant of the direction in which they are headed.

The only meaningful direction from either the Office of the Secretary of Defense (OSD) or the military services in the last few years was the Gansler memo that directed the use of the Software Engineering Institute's (SEI) Capability Maturity Model® (CMM®) Level 3 contractor organizations for Acquisition Category (ACAT) 1 systems. Do you know how many large-dollar (by that I mean $50 million or more) software intensive acquisitions are not ACAT 1? Virtually all Management Information System (MIS) and Command, Control, and Communications (C3) systems!

During the past two years, there has been a 5.5 percent annual growth in the cost of ACAT 1 programs due to cost and schedule estimating and engineering changes (sound like software?). Yet these programs have the most experienced DoD industry managers, and have a requirement for CMM Level 3. About two-thirds of DoD acquisition dollars are for programs below the ACAT 1 threshold for which there is currently no CMM requirement. It is my guess that these non-ACAT 1 programs are at least twice as bad as ACAT 1 programs – in other words, about $9 billion per year in cost growth associated with estimating and engineering problems, many of which are likely software related. In my opinion, they deserve more software management attention than results from the requirement to use best commercial practice.

CMM Maturity Reality

What is wrong with best commercial practice? It just does not exist among DoD contractors. It is a fantasy created by those who want to streamline acquisition, making it possible to cut the number of oversight personnel by reducing the opportunity for oversight. The best way to justify a hands-off attitude is to insist that contractors always do everything right!

There are more mature software organizations today. Virtually every large DoD contractor can boast at least one organization at CMM Level 4 or above, and several organizations at CMM Level 3. On the other hand, most DoD software is still being developed in less mature organizations – mainly because the program executive officer or program manager (PM) doesn't demand that the part of the company that will actually build the software be Level 3!

Many people used to tell me that the DoD needed to get on the dot-com bandwagon – those folks develop and deliver software fast. Yes, the Internet and the availability of Web browsers have fundamentally changed the environment within which even mission critical software is being developed. But instead of adapting proven software methods, the software research community has all but dropped its concerns with formal methods, peer reviews, clean-room processes, and other reliability techniques, including Ada, which was designed to promote reliability. Except for Barry Boehm at the University of Southern California, much of the academic community has more or less stopped investigating better ways of estimating system complexity and measuring software growth. Instead, invention of new user interfaces, new distributed computing architectures, and new (more flexible and less reliable) programming languages have been given top priority. The goals of reliable performance and pre-

dictable development costs have been largely ignored.

Wayt Gibbs, author of the 1994 *Scientific American* article, "Software's Chronic Crisis" told me: "It is tempting to argue that the lack of a disciplined approach to software development is the principal reason that so many dot-com ventures failed – and consumed such flabbergasting amounts of money as they failed."

That may be stretching it. Addle-brained business models clearly played a starring role as well. But it is interesting that, with very few exceptions, the dot-com startups embraced the programmer-as-master-craftsman model of development. That very old-fashioned model is still considered avant-garde in the open source movement. How many software startups in the past eight years have submitted to SEI evaluation or tried to achieve CMM Level 3? You won't need many fingers to count them all.

In addition to abandonment of formal methods, the Internet has also made obsolete the fortress model of security and the notion that an astute administrator can enforce central control of all components of a system. It is no longer realistic to dream of a mathematical certainty that each software component in a system is correct. In a world where network communication and user interface standards depend on change every 12-18 months, it does not make sense to lock down detailed requirements, spend a year proving them consistent, then spend two years building to spec.

A Move to Self-Sustaining Systems

There is a new move toward something called *autonomic computing*. This is a vague term that encompasses several lines of research aimed at taming complexity to achieve reliability. The approach is not *formal* but *organic*, i.e., find ways to build systems that can heal, that can achieve their mission despite bugs, equipment failures, and even sabotage. In other words, design systems that constantly monitor their own performance, that notice failures, can perform self-maintenance, and do not *crash* but degrade gradually. Some folks say these goals can be achieved in 10 to 20 years, but do not bet your paycheck on it.

Ironically, the problem facing much of industry is exactly the opposite. Wayt Gibbs told me the following: "Executives who woke up one day in 1997 to discover a time bomb in the form of Y2K bugs ticking inside their systems were forced to take a substantial hit to their bottom line. But an ironic consequence of Y2K is that many companies upgraded their affected systems in 1999 to work with or through the Internet and various fly-by-night standards and startup-built *solutions* that are now as obsolete as Algol. As a result, their systems are again ticking time bombs. Though they will not all fail at the same time, their failure will be just as unpredictable and much harder to fix."

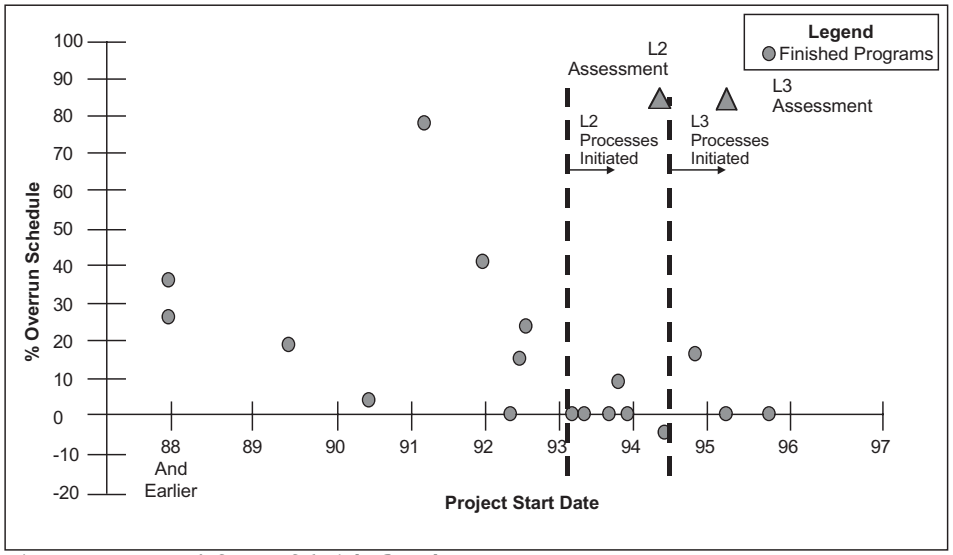


Figure 1: *Impact of SPI on Schedule Compliance*

The Missing Agenda

Some subjects have been notably missing from the plenary sessions of conferences during the past few years: software engineering, product line development, formal methods programming, and predictability.

In 1991, Paul Strassmann, then-director of Defense Information, said: "The No. 1 priority of the DoD, as I see it, is to convert its software technology capability from a cottage industry into a modern industrial method of production."

Guess what? That has not happened. Why not? Because this requires software engineering, which encompasses a set of three key elements – methods, tools, and procedures – that enable the manager to control the process of software development and provide the practitioner with a foundation for building high quality software in a productive manner.

The fundamental ingredient in a software engineering approach is the design of robust software architecture. Architecture does not refer to grouping and linkage of servers, routers, and PCs, but rather to the inherent design of the software itself – the identity of modules and their relationships, including the infrastructure, control, and data interfaces that permit software components to operate as a system.

I was told by an attendee at a DoD Systems Design Review several months ago that a contractor had described his proposed system as modular. That is a good architectural term, and it was accepted at face value. In fact the system, when looked at by the independent attendee, only had two modules. When this was brought to the government program manager's attention he said, "The contractor says it is modular. He's smarter than we are." This little incident underscores two facts: architecture was understood by neither the contractor nor by the government PM, and the probability of a successful software delivery is low.

All too often the DoD excuse for not requiring an architectural evaluation is that "requirements change rapidly – we can't afford to be locked into a specific architecture." Wrong. That is the very reason that attention should be given to architecture, so that changes to requirements can be accommodated easily.

I am told that SEI is still being asked to do Independent Technical Assessments of why a software acquisition has not produced the desired working software system. Why wait to ask SEI to come in to do a post-mortem and tell you how you screwed up? Why not ask them to come in first and review the Request for Proposal (RFP) and Statement of Work and, second, assist in evaluating the software engineering practices, software development environment, and architecture proposed in response to the RFP, and then afterwards assess the quality of the software engineering and development process? SEI is not cheap, but terminating a $100-million project for lack of satisfactory performance is not cheap either.

Interestingly, when one thinks about best commercial practice, there are two very different worlds out there. There is the government contractor world and there is the real commercial world – banks,

insurance companies, UPS, FedEx, Eckerd Drug, and Disney World. These companies developed their own software using the best available tools like Rational's software development environment. They did not pick the cheapest tools. They did not rely on commercial-off-the-shelf (COTS) or outside software developers – their software is their business. They consider that it provides them a competitive advantage. They want to control it, and they use the best tools available regardless of cost.

Architecture-Centered Approaches

Product line developments are also becoming increasingly commonplace in the true commercial world. The Swedish firm CelsiusTech was the first to exploit the benefits of a product line architecture approach to software application development back in the late 1980s. There are now a number of well-known firms who are using an architecture-centered approach: Nokia, Motorola, Hewlett-Packard, Philips, Cummins Engine, and (believe it or not) one government application at the National Reconnaissance Office (NRO).

The NRO is enjoying a 50 percent reduction in overall cost and schedule, and nearly tenfold reductions in defects and development personnel. Let me list the most common and relevant-to-DoD reasons that these companies give for adopting the architecture-centered product line approach to software development:

- Large-scale productivity gains.
- Improved time-to-market = field weapons faster.
- Improved product quality.
- Increased customer satisfaction = warfighter satisfaction.
- Enabled mass customization.
- Compensated for inability to hire software engineers.

These companies and the NRO do have *best practices* but are not yet widely recognized as such. Frankly, it will take DoD program executive officers (PEOs) (not program managers) and their overseers, the service acquisition executives, and especially the DoD comptroller and program analysis and evaluation folks, to recognize and direct the product line approach to software development and acquisition. (Unfortunately, these folks are not known for their software acumen.)

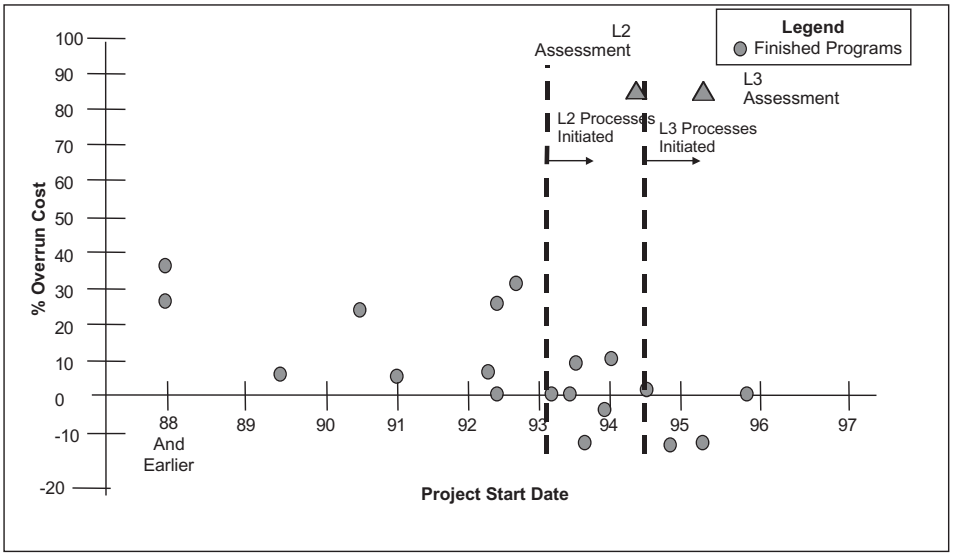
The major impediment to the product line development approach, aside from ignorance of its benefits, are cultural, organizational, and, especially, the DoD's stovepipe approach to budgeting and funding. The DoD has many potential software product lines. None of them have been built largely for *political* and stovepipe budgeting reasons. As a result, development and fielding of defense systems continues to take longer, cost more, and lack predictable quality. Product lines require strategic investment, which appears to be outside the DoD comptroller and acquisition communities' frames of reference. Yet, it is the DoD that most often uses the term *strategic*.

Cummins' Engine Company used to take a year or more to bring software for a new engine to the test lab – now they can do it in less than a week! I strongly recommend that readers obtain a copy of a new book, "Software Product Lines," just published by Addison-Wesley. The authors are from SEI. Sadly, with the exception of the NRO, it appears that the readers are mainly from commercial organizations.

I work for one, but let me tell you, although government contractors are commercial organizations, they do not have an identifiable best commercial practice. They basically provide what the government asks for. The only reason many of them can now boast of having at least a few SEI maturity level organizations is because 10 years ago, many government RFPs required a Software Capability Evaluation (SCE) as a condition of bidding. In fact, sad to say, many contractors today put satisfactory CMM credentials in their proposals but then perform the work with an organization that could not spell CMM.

Why does the government let this happen? Why aren't there SCEs anymore? Do they take too long and cost too much? Is it better to make a quick award, and then, down the line, accept an inferior product or terminate for convenience? Too often what the government has been asking for is COTS. How many failures of COTS-based acquisitions have there been over the past decade? Too many!

Figure 2: *Impact of SEI on Cost Compliance*



What is Best Commercial Practice?

Best commercial practice is not eliminating all software smarts in government and relying 100 percent on contractors to deliver good software. Best commercial practice is what real commercial companies are doing. They have in-house software expertise, they use a robust software development environment, and they base their software development on sound software architecture. It is no secret that Rational and their competitors have a growing market in the commercial world and a shrinking market in the government. I am not suggesting that the government can or should develop software in-house. I am strongly suggesting that the government needs enough in-house software expertise to know what it is buying. It is still true that you "get what you pay for."

Watts Humphrey recently published "Winning with Software - An Executive Strategy." This book is directed primarily at executives of commercial companies. But every DoD acquisition executive, PEO and PM needs to read and understand its simple message: Software projects rarely fail for technical reasons; invariably, the problem is poor management.

Watts poses two interesting questions buttressed by numerous examples: "Why do competent software professionals agree to dates when they have no idea how to meet them?" "Why do executives accept schedule commitments when the engineers offer no evidence that they can

meet these commitments?" He asserts that management's undisciplined approach to commitment contributes to every one of the five most common causes of project failure:

- Unrealistic schedules.
- Inappropriate staffing.
- Changing requirements.
- Poor quality work.
- Believing in magic.

What is Formal Methods Programming (FMP)? Basically, FMP is all of the above rolled together: sound management, established engineering processes, robust software development environment, model-based architecture, and a reliable programming language. Peter Amey of Praxis Critical Systems said in March 2002 CROSSTALK: "There is now compelling evidence that development methods that focus on bug prevention rather than bug detection can raise quality and save time and money." He went on to say that a key ingredient is the use of unambiguous programming languages that allow rigorous analysis early in the development process.

I was an early and vocal advocate of Ada, primarily because, unlike other languages, it enforces basic software engineering practice. Amey's article describes the use of a subset of Ada known as SPARK, which he says requires software writers to think carefully and express themselves clearly; otherwise, lack of precision is exposed by SPARK Examiner. He said there were significant savings in using SPARK in a critical avionics program, including an 80 percent reduction in formal test costs. Unfortunately, this is an isolated DoD example.

Finally, let me say a word about predictability. Predictability is the only metric that warfighters care about. The question is, how can we make the warfighters (including the PEOs and PMs charged with delivering the needed capabilities) know that you cannot just buy software as a product off the showroom floor? There must be an understanding of the software-engineering paradigm. More than this, to be assured of getting software that works on a predictable schedule and at predictable cost requires that someone in government enunciate requirements. Or else, competitors will lowball price and set an unrealistically fast schedule and be awarded the contract. To perform at low cost means no robust software development environment, no time and effort devoted to creating a valid software architecture, and probably means cheap people. Sufficient process guidance must be given to assure that contractors all bid at the same capability level. Government should be explicit about its need for architecture, a robust software development environment, and perhaps even the requirement for a language like SPARK. At a minimum, it needs to specify at least CMM Level 3. As the figures illustrate, (see pages 23, 24) at Level 1 virtually all projects have cost and schedule overruns, whereas at Level 3 virtually all projects are on target. Regarding defect rates and cost ($) per source line of code, there is substantial improvement on the order of 20 percent to 50 percent. It really is true that "you get what you pay for." If you want it cheap, you'll get it cheap – but the software may not work in the manner envisioned, if it will work at all.

As for CMMI, I believe it is very important for the embedded world… but it would be a gross mistake to discontinue support of the Software CMM.

Are there best commercial practices? You bet! The banks, insurance companies and other truly commercial enterprises have them. Not because of some automatic external happenstance, but because their senior managers have had the moxie to realize that it takes money to make money, and that it takes software expertise to develop or acquire software. The government needs to *go and do likewise*. Otherwise, the decade of 2000 will likely not show any lessening of the software crisis that has carried over from the 1990s.◆

About the Author

Lloyd K. Mosemann II, is the senior vice president of Corporate Development for Science Applications International Corporation (SAIC). Formerly, for almost 25 years, Mosemann was deputy assistant secretary of the Air Force for Communications, Computers, and Logistics. During this time, he chartered and guided the Air Force's Software Technology Support Center and sponsored its annual Software Technology Conference. Prior to that, Mosemann spent 11 years with the Navy. He has a bachelor's and master's degree from the University of Chicago, and has received two Presidential Meritorious Rank Awards, five Air Force Exceptional Service Medals, among other awards.

Science Applications International Corp.
E-mail: lloyd.k.mosemann.II@saic.com

Best Value Formula

David P. Quinn
National Security Agency

A constant concern when preparing to release a Request for Proposal is that one bidder will throw things completely out of whack by "lowballing." This means the bidder will bid extremely low, willingly incurring a loss in most cases, just to get the job and position themselves for future contracts. Because the bidder offers such a low price for the contract, limitations in their technical and management proposals get overlooked. This article proposes the Best Value Formula for reducing the impact of lowballing proposals by tying the price offered more closely with the technical and management proposals of a bidder. A high-level description of the proposal evaluation process is given as context. The Best Value Formula is defined and explained. Finally, examples of bids are given to demonstrate the effectiveness of the Best Value Formula.

Our organization is in the process of preparing for a firm, fixed-price contract to perform a set of concept studies. The results of these studies will be used as input in a development and integration contract. The concept studies contract is not considered a lucrative contract. The prize is actually the development and integration contract. Everyone believes the winner of the concept studies contract will have an inside track on the more lucrative contract.

When developing the proposal evaluation criteria, we were haunted by the fact that we could only take the weight of the price factor so low (30 percent) without requiring a General Accounting Office audit of the contract bidders. The weight for the price factor looked relatively high, especially for a fixed-price contract. Our fear was that one of the bidders would bid incredibly low just to get in position for the follow-on contract, and that the technical and management factors would become worthless at that point.

This is not to say that we would not have welcomed a very low price for a very good technical and management proposal. Ideally, this is what everyone wants. We just wanted assurances that this would be the case and that a poor proposal did not win just because it was priced excessively low.

Evaluating Proposals

While some people may think that price is the only factor in determining who wins a government contract, it is not. Generally, there are four major factors when evaluating contracts: technical approach, management approach, past performance on similar contracts, and price. Each major factor is assigned a weight such that the sum of the weights equals 100. The assigned weights allow for a greater emphasis to be placed on one major factor over another. An even-weight distribution would have weights of 30 for technical, 30 for management, 10 for past performance, and 30 for price. For our Request for Proposal (RFP), the assigned weights were 60 for technical, 10 for management, and 30 for price. Past performance was made a pass/fail factor with no weight.

Table 1: *Example of Technical Score*

Factor/Subfactor	Weight	Rating	Score
Technical	60	88.3	53
Trade Studies	20	85	17.0
Architectures	15	90	13.5
Innovation	25	90	22.5

Each major factor may have one or more subfactors that comprise the major factor. For instance, management may have subfactors of project management and key personnel. Each subfactor is weighted and scored individually. For our RFP, the technical factor had subfactors of trade studies, architectures, and innovation with weights of 20, 15, and 25, respectively.

> *"The desirable position for the government is to find a way that directly considers the price bid with the technical and management capability so that price is not the true deciding factor."*

When evaluating proposals, a defined set of criteria for each subfactor is rated. The rating is done as a percentage of a subfactor and has a description associated with it. The usual rating scale is as follows:

- Excellent 90-100
- Good 80-89
- Acceptable 70-79
- Marginal 60-69
- Unacceptable 0-59

Unacceptable ratings are based on completely missing one of the criteria for a subfactor or major factor. Marginal means that there are faults in the proposal against certain criteria but the criteria are addressed. Acceptable means that the criteria are met. Ratings above acceptable indicate that the proposal had some additional information that helped it stand out.

The score for a factor is therefore defined as the sum of the scores of the subfactors. The score of the subfactor is the rating times the weight. Using our technical factor as an example, a sample scoring is shown in Table 1. The final score for a proposal evaluation is the sum of the scores for the major factors. In most instances, the final score formula looks like this:

Final Score = Technical Score + Management Score + Price Score

The highest final score is considered the contract winner. To select a bidder that did not receive the highest score requires lots of extra paperwork. In the case of a similar contract to ours, 500 pages of justification were generated to not pick the highest score.

Cost as a Factor

One factor that is not rated on the scale in Table 1 is price. Cost simply indicates what the vendor will charge for its services. Therefore, all price proposals are assumed to be acceptable.

There is a very generic formula used when determining price as a factor for most contract proposals. All the proposals are received and the lowest price of all the proposals becomes the standard by which all the proposals are evaluated. One at a time, each proposal is evaluated by taking the lowest proposal price and dividing it by the price of the proposal being evaluated. That fraction is then multiplied by the weight of the price factor for the price

score. The formula looks like this:

Price Score = Price Weight * (Lowest Price/Current Proposal Price)

In theory, this is not bad. It works best when the proposed prices are all in the same neighborhood. For instance, everyone bids in the $8 million to $10 million range. However, when theory meets reality, reality tends to win.

If one bidder really sends in a low price, all the other proposals pay the consequence. If three bids are in the $8 million to $10 million range but a fourth bid comes in at $4 million, the other proposals lose almost half the price factor points immediately. It requires that the $4 million proposal be deemed unacceptable for its technical or management proposal in order to lose and not have any impact on determining the contract winner.

Table 2 is an example of a bidder trying to get a contract based on an extremely low bid. Due to the extremely low bid price of bidder 5, bidders 1 through 4 lost at least half the number of price points available. The impact really is that if bidders 1 through 4 received ratings of 100 on each factor, the best overall score they could get is 85.

Examples of Price Impact on Contract Award

It is important to see what this looks like in terms of comparative bids on a contract. Table 3 shows five bidders' proposals on a contract, with two of the bidders trying to lowball the other bidders. Past performance will be pass/fail so no weighted scores are needed for it.

As the final scores show, the order of award follows the order of price from least to most (i.e., bidder 5, bidder 4, bidder 3, bidder 1, then bidder 2). Bidder 5 was able to win the contract, despite having a barely adequate proposal, by lowballing the bid. Obviously, this does not give the government the best value for its money and perpetuates the stereotype that the lowest bid always wins. The government's only hope is that the bidder fails the past performance factor.

Finding the Real Best Value

The desirable position for the government is to find a way that directly considers the price bid with the technical and management capability so that price is not the true deciding factor. In essence, the government should receive the best value for its investment by ensuring the price is proportionate to the technical and management proposals.

This actually makes the price evaluation more consistent with the rest of the proposal evaluation process. Technical and management proposals are evaluated independent of the other bidders' technical and management proposals. Great strides are taken to ensure that one proposal does not influence the rating of another proposal. However, the price proposal is directly evaluated against the other bidders' price proposal. The price evaluation needs to move away from strictly looking at comparisons among proposals.

To address price in relation to technical and management proposals, the weight of the price factor should be adjusted based on the scores of the technical and management proposals. If you add the technical and management scores and divide that sum by the sum of the technical and management weights, a Best Value Ratio (BVR) is created. The BVR is multiplied by the price factor weight to get the Best Value Factor (BVF) for the proposal. The BVF is then substituted for the price weight to calculate the price score. The formulas for this series of computations are as follows:

Best Value Ratio = (Technical Score + Management Score) / (Technical Weight + Management Weight)

Best Value Factor = Best Value Ratio * Price Weight

Factor	Bid 1	Bid 2	Bid 3	Bid 4	Bid 5
Bid Price	9	10	9	8	4
Lowest Price	4	4	4	4	4
Price Weight	30	30	30	30	30
Price Score	13.3	12	13.3	15	30

Table 2: *Impact of a Lowball Bid*

Best Value Score (or Price Score) = Best Value Factor * (Lowest Bid / Current Price Being Evaluated)

Table 4 shows the results of applying the BVF to the bids in Table 3. The BVF changed the order of bidders to better reflect the government's desires. Assuming all bidders pass the past performance criterion, bid 3 would be awarded the contract since its strong technical and management proposals had little impact on its competitive price. Bid 5's attempt to lowball the bid goes unrewarded as their weak technical and management proposals weakened the impact of their low price. The bid that provides the best value is identified and rewarded.

Whither Go Past Performance

The examples above were all based on the assumption that past performance is a pass/fail factor, and it does not have any weight associated with it. If past performance is a rated factor with an associated weight, it is up to the acquisition organiza-

Table 3: *Example of Proposal Scores*

Factor	Bid 1	Bid 2	Bid 3	Bid 4	Bid 5
Technical Weight	40	40	40	40	40
Technical Rating	85%	90%	90%	80%	70%
Technical Score	34	36	36	32	28
Management Weight	30	30	30	30	30
Management Rating	90%	90%	90%	80%	70%
Management Score	27	27	27	24	21
Price Weight	30	30	30	30	30
Price Bid	7	10	6	4	3
Lowest Price Bid	3	3	3	3	3
Price Score	12.9	9.0	15.0	22.5	30.0
Final Score	**73.9**	**72.0**	**78.0**	**78.5**	**79.0**

Table 4: *Example of Best Value Formula Results*

Factor	Bid 1	Bid 2	Bid 3	Bid 4	Bid 5
Technical Weight	40	40	40	40	40
Technical Rating	85%	90%	90%	80%	70%
Technical Score	34	36	36	32	28
Management Weight	30	30	30	30	30
Management Rating	90%	90%	90%	80%	70%
Management Score	27	27	27	24	21
Price Weight	30	30	30	30	30
Price Bid	7	10	6	4	3
Lowest Price Bid	3	3	3	3	3
Price Score	12.9	9.0	15.0	22.5	30.0
Old Final Score	**73.9**	**72.0**	**78.0**	**78.5**	**79.0**
Best Value Ratio	.8714	.9000	.9000	.8000	.7000
Best Value Factor	26.1	27.0	27.0	24.0	21.0
Best Value Score	11.2	8.1	13.5	18.0	21.0
New Final Score	**72.2**	**71.1**	**76.5**	**74.0**	**70.0**

Setting Up a Spreadsheet

A formula is only valuable if it is applied properly, and it can be automated through a spreadsheet. Below are helpful hints on how to set up a spreadsheet to calculate best value variables. Included are examples of spreadsheet formulas that can be used and the points in the source selection process when a certain value can be entered.

Three assumptions will be made. First, past performance is assumed to be pass/fail or a subfactor under the management factor. Second, the first row of the spreadsheet will be used as a header row to identify the various bidders. Third, the first column in the spreadsheet will be used for variable names to identify where to assign values.

	A	B	C	D	E
1	**Variable Name**	**Values**	**Describes**	**Example**	**Available After**
2	Technical Weight	Constant	Weight assigned to the technical proposal.	40	Source selection criteria released.
3	Technical Rating	0 to 100	Percent number representing the adjectival rating for the technical proposal.	84%	Source selection team consensus.
4	Technical Score	0 to Technical Weight	Normalized score for technical proposal computed by multiplying values in rows 2 and 3.	= B2 x B3	Technical rating is entered.
5	Management Weight	Constant	Weight assigned to the management proposal.	30	Source selection criteria released.
6	Management Rating	0 to 100	Percent number representing the adjectival rating for the management proposal.	78%	Source selection team consensus.
7	Management Score	0 to Management Weight	Normalized score for management proposal computed by multiplying values in rows 5 and 6.	= B5 x B6	Management rating is entered.
8	Price Weight	Constant	Weight assigned to the price proposal.	30	Source selection criteria released.
9	Price Bid	Constant	Price bid by vendor; could be rounded to nearest thousand if desired.	$9,455	When proposals are received.
10	Lowest Price Bid	Lowest Price of all Bids	Lowest price of all bids received.	Minimum of row 9	After prices are entered.
11	Best Value Ratio	0 to 1	Computed by adding rows 4 and 7 then dividing by sum of rows 2 and 5.	= (B4 + B7) / (B2 + B5)	Computed when technical and management ratings are entered.
12	Best Value Factor	0 to Price Weight	Normalized score for price proposal computed by multiplying values in rows 8 and 11.	= B8 x B11	Computed when technical and management ratings are entered.
13	Best Value Score	0 to Price Weight	Price value in terms of best value.	= B12 x (B10 / B9)	Computed when technical and management ratings are entered.
14	Final Score	0 to 100	Total of technical, management, and price scores.	= B4 + B7 + B13	Computed when technical and management ratings are entered.

tion to determine if past performance scores should be part of the BVF. If it is decided that past performance will be part of the BVF, the past performance score should be added to the technical and management scores in the BVR. Additionally, the past performance weight should be added to the technical and management weights in the ratio. The BVR would then look like this:

Best Value Ratio = (Technical Score + Management Score + Past Performance Score) / (Technical Weight + Management Weight + Past Performance Weight)

Punishment or Reward?

Is a bidder being penalized twice for a weak technical or management proposal? As the example above shows, all the bidders were deemed acceptable. It is hard to call applying their technical and management scores to their price proposal a punishment. However, a bidder that provides an *excellent* proposal should be rewarded. The BVF rewards bidders who have stronger proposals.

More importantly, the question is this: Is it fair to punish the government with a less-qualified bidder just because they had the lowest price? The BVF is a method for reflecting the government's true best interest. It is meant to help quantify where the government gets the best technical and management implementation for its money.

Scores Are Just an Aid

In any proposal evaluation, the awarded scores cannot be the sole basis for final judgment. Other factors such as price realism and fit with government oversight practices are considerations. The BVF is an aid that provides a more appropriate order to the bidders but it is not a substitute for sound reasoning. The final award requires written justification stating what makes one bid superior to another.

Validating the BVF

A program similar to ours just completed awarding three contracts to conduct concept studies. There were four bidders and one of them tried to lowball the bid, significantly. The lowball bid had the worst technical and management proposals but they had the highest score based on their low price. It required 500 pages of documentation to support not awarding one of the three contracts to this bidder.

The scores from this program's evaluation were entered into the BVF. The lowball bid ended up having the lowest score of the four bids. The BVF placed the bid in an order that best represented best value to the government.

Conclusion

When going to contract, the government should have a tool that alleviates the concern that a bidder is going to throw the entire acquisition out of line by focusing on price vs. a sound technical and management proposal. The current method for determining the impact of price is based on a comparison between bids. Price needs to be considered in direct correlation with technical and management proposals. The BVF considers price with relation to the other factors. It does a much better job of focusing the proposal evaluation process away from price and towards a more complete picture of the proposal.◆

About the Author

David P. Quinn was a senior computer scientist for the National Security Agency. He has 19 years of software and systems engineering experience, focusing the last eight years on process improvement. He is a Software Engineering Institute-certified Lead Assessor and served on the Capability Maturity Model® Advisory Board.

Sensible Process, LLC
30 Quail Ridge Road
Hanover, PA 17331
Phone: (717) 451-2149
E-mail:dquinn@sensibleprocess.com

Platform Independent Tactical Data Entry Devices

Lt. Col. Kenneth L. Alford
U.S. Military Academy

Maj. John Surdu, 2nd Lt. Gene Yu, 2nd Lt. William Herrington, 2nd Lt. Charles Maranich

As the Army becomes increasingly digitized, the need for technically advanced equipment is imperative. The project in this article investigates the implementation of platform-independent software to run on common hand-held units. The advantage of a "software first" approach is that it frees the Army from over-reliance on vendors/integrators. The research described in this article is an attempt at a platform and vendor independent data input device for Field Artillery forward observers. The article can be viewed in its entirety at <www.stsc.hill.af.mil>.

Web Sites

Software Technology Support Center

www.stsc.hill.af.mil

The Software Technology Support Center is an Air Force organization established to help other U.S. government organizations identify, evaluate, and adopt technologies to improve the quality of their software products, efficiency in producing them, and their ability to accurately predict the cost and schedule of their delivery.

Software Program Managers Network

www.spmn.com

The Software Program Managers Network (SPMN) is sponsored by the Deputy Under Secretary of Defense for Science and Technology, Software Intensive Systems Directorate. It seeks out proven industry and government software best practices and convey them to managers of large-scale DoD software-intensive acquisition programs. SPMN provides consulting, on-site program assessments, project risk assessments, software tools, guidebooks, and specialized *hands-on* training.

Defense Acquisition Deskbook

www.web2.deskbook.osd.mil/5000Model.asp

The Defense Acquisition Deskbook is sponsored by the Office of the Under Secretary of Defense. It functions as a hub for the Department of Defense's acquisition center, including the DoD 5000 Model, quick links, reference library, ask a professor, special interest items, education and training, and more.

Software Cost Estimation Web Site

www.ecfc.u-net.com/cost/index.htm

The Software Cost Estimation Web site presents a review of current cost estimation techniques to help industry and academia choose the appropriate methods when preparing software cost estimates. The site covers both traditional and state-of-the-art methods identifying advantages and disadvantages of each and the underlying aspects in preparing cost estimates. The site also provides links to other software cost estimation sites that are involved in this area and details the research that has been undertaken at Bournemouth University.

Software Engineering Institute

www.sei.cmu.edu

The Software Engineering Institute (SEI) is a federally funded research and development center sponsored by the Department of Defense to provide leadership in advancing the state of the practice of software engineering to improve the quality of systems that depend on software. SEI helps organizations and individuals improve their software engineering management practices. The site features complete information on models the SEI is currently involved in developing, expanding, or maintaining, including the Capability Maturity Model Integration, Capability Maturity Model for Software, Software Acquisition Capability Maturity Model, Systems Engineering Capability Maturity Model, and more.

The Software Productivity Consortium

www.software.org/default.asp

The Software Productivity Consortium is a nonprofit partnership of industry, government, and academia. It develops processes, methods, tools, and supporting services to help members and affiliates build high-quality, component-based systems, and continuously advance their systems and software engineering maturity pursuant to the guidelines of all of the major process and quality frameworks. Its Technical Program builds on current best practices and information technologies to create project-ready processes, methods, training, tools, and supporting services for systems and software development.

Practical Software and Systems Measurement Support Center

www.psmsc.com

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

Raiders of Best Value - Search for the Holy Acquisition Grail

While attending college in the northern wilderness of Utah at Utah State University, one reprieve from the pursuit of knowledge that the local constable considered legal was to go to the movies. Although the crowds flocked to blockbuster movies like *Star Wars* and *Jaws,* the movie experience I cherish the most comes from a guy wearing a dusty fedora.

I was pouring over electromagnetism when a friend asked if I wanted to go to a movie about pirates searching for Noah's lost dinghy. Interesting concept, I thought. "Who is in the movie?"

"I think Han Solo," he replied.

"Anybody else I would know?" I asked.

"No, but the *Jaws* guy and the *Star Wars* guy got together and created it," he said.

"Will I have to wait in line?"

"I doubt it," he said. "There hasn't been much hype for the movie, and it was released yesterday."

We arrived at the theatre two minutes before show time, walked right in, and confiscated prime seats. The opening scene absolutely captivated my attention. With sweat beading on his brow, Indiana Jones replaced a golden idol with a bag of sand of equal weight and then, all hell broke loose. There was a giant rolling boulder, breakaway floor, spring loaded spikes, headhunters, darts, arrows, and … well you know the story. In five minutes I felt like I had watched an entire movie.

That started my fascination with the persistent man who often ended up short of his intention. While most reminisce of Indiana Jones as a hero, if you think about it, he often fell short of his goal. The ark ended up in a massive government warehouse next to the Ada mandate, only one Sankara stone out of four was recovered, and the Holy Grail was lost in an earthquake – didn't he learn anything from Monty Python? Time after time, Jones has a hard time retaining his treasures.

Indiana's honorable futility reminds me of government acquisition, software or otherwise, and the search for the best way to acquire goods and services. While the acquisition process rarely entails mortal peril, it is as unpredictable as a ride through the Temple of Doom and in the end, whether buyer or seller, you feel like the high priest of the temple has reached in and ripped out your heart. The government's intentions and theories are admirable but their adventures, like Indy's, often run astray.

It amazes me that the same government that owes its prosperity, growth, and wealth to Adam Smith's invisible hand of the market bites the very hand that feeds it. When the government becomes the consumer, it does everything but trust the market, resembling Python's "so-called Arthur-king," bumbling across the landscape looking for the Holy Grail of acquisition best value.

What is best value, and where did it come from? I propose that the concept of best value rose out of the government's lack of trust in the market and its great mediator – money – also known by its acquisition name, *price*. After experiencing $5,000 hammers and the "lowball bid," (the wrecking ball of the acquisition process that contractors use to get their foot in the door), they must have concluded that price alone would not suffice. Smith must have been wrong and the government, with its vast knowledge and resources, would correct the volatility of the market.

Ignoring their role in the debacle, the government turned to acquisition alchemists who were confident they could add elements to price that would produce best-value gold. The most popular ingredients were technical approach, management approach, and a pinch of past performance. Has it worked? Not in my experience. Instead they created a gaming system that rivals Las Vegas.

Take technical approach. In order to evaluate technical approach, you have to articulate what you want and understand the differences and nuances in the technologies that could meet that need. In reality, most Request for Proposals (RFP) and solicitations are harder to decipher than teenage chat room dialogue. They are seldom evaluated by anyone who could recognize a valid technical approach from a Tom Clancy novel. This leads to contractors who forego creativity for mimicking. The worst-kept secret in developing proposals is that contractors are taking technical requirements from the RFP and feeding them back to evaluators, like a cheap pet-store parrot asking for a cracker.

Next comes management approach. In this case, leave the parrot at home and pick up a standard management book or better yet, hire a PMI-certified consultant. Hit all the key words and throw in a unique buzzword that demonstrates you are on the cutting edge of management. Business experts can't agree on the best management approach, but government evaluators can?

Finally, past performance references, like job references, are a dying breed: Who's not going to give you anything but references that will provide a shining review of your company?

The next step is to statistically aggregate, interpret, and manipulate the evaluation data and come up with a remedy. Enter weighting, algorithms, and a whole lot of hand-waving. So now we are spinning evaluation data more than Michael Jackson hits in the '80s. If we keep it up, we will have more evaluation criteria and statistical algorithms than advertisement patches on a NASCAR driver's jumpsuit.

It reminds me of Herbert's father in Monty Python's *Holy Grail*. He built a castle in the swamp and it sank. He built a second castle in the swamp, and it sank too. He built a third and it burned down, fell over, and then sank into the swamp, and he can't understand why Herbert doesn't want to inherit a fourth castle. When will we stop chasing best value through the swamp and return to our roots – price?

As Smith points out, money or price is not just the simplistic instrument of commerce, but in a free market it is "the measure of value." Does it give you best value? No. It doesn't protect those who don't do their homework. Too many variables are involved to assure best value, but in an unconstrained market, a wise consumer can get good value.

Given experience with both, I'll take the market's good value over the government's best value.

— Gary Petersen
Shim Enterprise, Inc.

Don't Leave Your Software Acquisition to Chance

The Right Partner Makes All the Difference

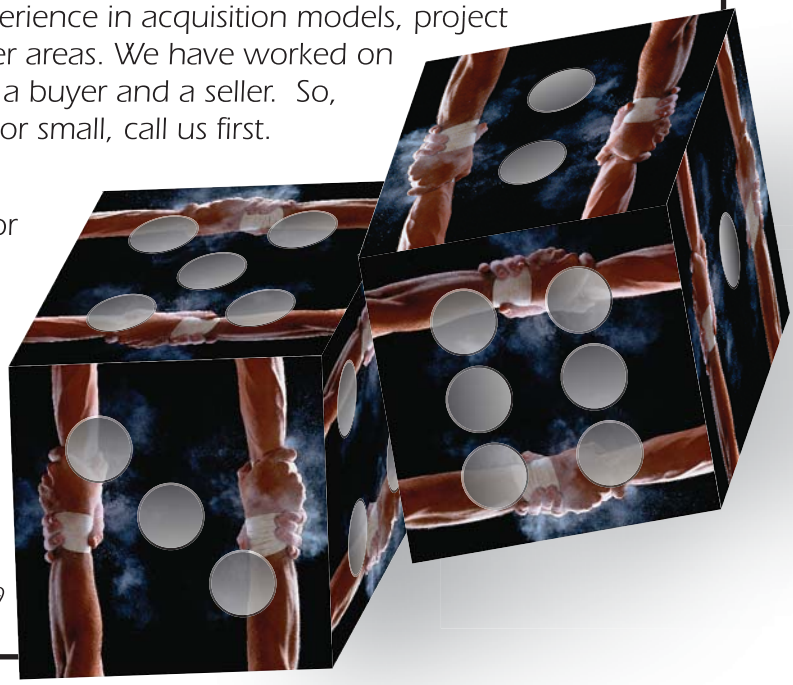
STSC Acquisition Services Can Help You Make the Safe Bet, Every Time:

- Acquisition Process Assessments
- Independent Expert Program Reviews
- Software Acquisition Capability Maturity Model Instruction
- Program Management/Oversight Assistance
- Risk Management
- Requirements Management
- Verification and Validation
- Documentation Review
- Just-In-Time Services

We have many years of experience in acquisition models, project management, and perimeter areas. We have worked on both sides of the fence – as a buyer and a seller. So, whether your project is big or small, call us first. We can help.

If you have any questions, or require more information, please contact us at the Software Technology Support Center.

OO-ALC/TISE
7278 4th Street
Hill AFB, UT 84056
801 775 5555 DSN 775 5555
Fax 801 777 8069 DSN 777 8069
www.stsc.hill.af.mil



Sponsored by the Computer Resources Support Improvement Program (CRSIP)

Published by the Software Technology Support Center (STSC)

CrossTalk / TISE
7278 4th Street
Bldg. 100
Hill AFB, UT 84056-5205

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