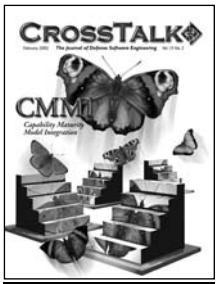
CROSSTALK

February 2002 The Journal of Defense Software Engineering Vol. 15 No. 2

CMMI

Capability Maturity Model Integration

Capability Maturity Model Integration



ON THE COVER

Kent Bingham, Digital Illustration and Design, is a self-taught graphic artist/designer who free-lances print and Web design projects.

Software Engineering Technology

Field Report

Departments

CROSSTALK

SPONSOR	*Lt. Col. Glenn A. Palmer*
PUBLISHER	*Tracy Stauder*
ASSOCIATE PUBLISHER	*Elizabeth Starrett*
MANAGING EDITOR	*Pamela Bowers*
ARTICLE COORDINATOR	*Nicole Kentta*
CREATIVE SERVICES COORDINATOR	*Janna Kay Jensen*
PHONE	(801) 586-0095
FAX	(801) 777-5633
E-MAIL	crosstalk.staff@hill.af.mil
CROSSTALK ONLINE	www.stsc.hill.af.mil/crosstalk/crostalk.html
CRSIP ONLINE	www.crsip.hill.af.mil

Subscriptions: Send correspondence concerning subscriptions and changes of address to the following address. You may e-mail or use the form on p. 24.

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.

Updated CMMI Focuses on Broader Usage in Industry and Government

This issue of CROSSTALK is the second dedicated to the Capability Maturity Model® Integration℠ (CMMI℠) project. Readers new to CMMI will find it useful to first review the CMMI articles found in the July 2000 CROSSTALK. Back issues can be found online at the Software Technology Support Center (STSC) Web site <www.stsc.hill.af.mil>. After more than an additional year of comments and piloting, CMMI Version 1.1 was released in January 2002. The focus has now turned to "implementation" of CMMI into broader usage in industry and the government. Our Air Force program office (Computer Resources Support Improvement Program) remains dedicated to the successful introduction of CMMI into government offices throughout the Department of Defense and other government agencies.

I hope that most people now recognize that CMMI is a cooperative endeavor between government, industry, and the Software Engineering Institute. The goal of the CMMI project starting in early 1998 has been to merge several separate discipline process improvement models into a single framework that can be used as a basis for a common approach to process improvement for both systems and software engineering. It is also expected that CMMI will provide for new disciplines to be added with relative ease. It is anticipated that CMMI Version 1.1 will prove to be as stable and long-lived as the source model for software that preceded it.

The first CMMI Technology Conference and User Group was held at the Denver Technology Center, Nov. 13-15, 2001. More than 300 individuals met to share lessons learned in the early implementations of CMMI. Full details of the presentations from this conference can be downloaded from the National Defense Industrial Association Conferences Web site: <www.dtic.mil/ndia/2001cmmi/2001cmmi.html>.

In this issue of CROSSTALK, CMMI Project Manager Mike Phillips describes in *CMMI Version 1.1: What Has Changed?* the key changes that have occurred with the release of CMMI Version 1.1. One of the biggest concerns voiced about the new CMMI model has been the length of time required for appraisals using this larger, combined model. Phillips also describes some of the discoveries that have been made from eight pilot appraisals accomplished in the Phase II Pilot Program, including improved efficiency in the execution of the appraisal method. In *CMMI Appraisal Methodologies: Choosing What Is Right for You*, Ilene Minnich discusses some alternatives to using the full class A Standard CMMI Appraisal Method for Process Improvement (SCAMPI). She also stresses that "the keys to success are education, preparation, and pre-work."

As mentioned above, one of the goals was to make the CMMI framework easily expandable for new discipline investigations. Donald R. Michels and Bonnie Bollinger from Warner Robins Air Logistics Center review the success that they have achieved by piloting new acquisition process areas in *Transitioning From SA-CMM to CMMI in the Special Operations Forces Systems Program Office.*

Consultant Winifred Menezes discusses in *To CMMI or Not to CMMI: Issues to Think About* some ideas for transitioning to CMMI from several different perspectives. In *How Function Points Support the Capability Maturity Model Integration*, Barbara Emmons and Carol Dekkers discuss the tie between CMMI process areas and Function Point Analysis. They indicate that this tie is not well known, but make a compelling case that a direct connection does exist.

Also included is an article by Sarah Sheard, *How Do I Make My Organization Comply With Yet Another New Model?* Sheard believes that complying with a new capability model is much easier than starting fresh if the organization already complies with another model. Lastly an article by Edgar Dalrymple, *U.S. Army Develops High Quality, Extremely Low Cost Digital Message Parser*, details how the Software Engineering Directorate, by virtue of their development process, successfully provided a key technology to support one of the Army's most significant organizational goals: interoperability via digitization.

Space limitations in this issue forced an article by Suzanne Garcia of the Software Engineering Institute, *Are You Prepared for CMMI?*, to be published in next month's issue. Garcia talks about how applying technology adoption concepts can smooth the CMMI adoption process considerably.

I hope that these articles, as well as additional ones about CMMI usage over the coming year, will provide incentive for your organization to get started or continue in your voyage to migrate to CMMI. For more detailed information on the CMMI project visit the SEI Web site at <www.sei.cmu.edu/cmmi>. For assistance in understanding and getting started with CMMI, also visit the STSC Web site at <www.stsc.hill.af.mil>.

H. Bruce Allgood
Deputy Director, Computer Resources Support Improvement Program

CMMI Version 1.1: What Has Changed?

Mike Phillips
Software Engineering Institute

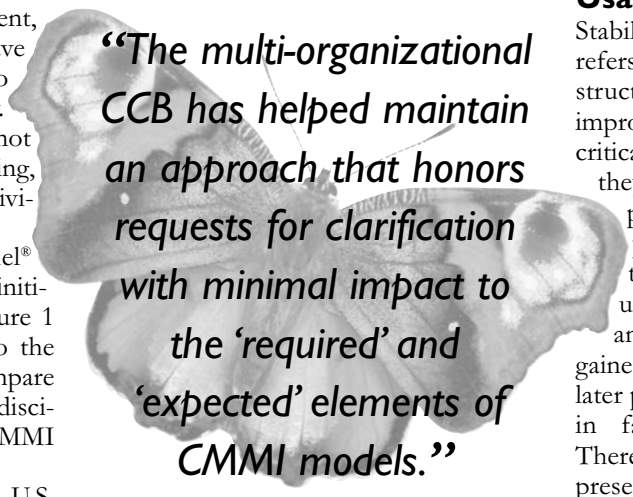
With Capability Maturity Model® Integration℠ (CMMI℠) V1.1, we have improved the "initial use" version provided to the community in the fall of 2000. The purpose of this article is to describe the key changes to the CMMI product suite, with some of the discoveries we made along the way.

Since 1991, Capability Maturity Models® (CMM®) have been developed for a variety of disciplines, including systems engineering, software engineering, software acquisition, work-force practices, and integrated product and process development. These models have been valuable to many organizations; however, the use of multiple models has been expensive and complicated. Organizations that want to pursue process improvement across disciplines have had to cope with differences in model architecture, content, and approach. These differences have limited these organizations' ability to focus their improvement successfully. Applying multiple models that are not integrated is costly in terms of training, assessments, and improvement activities.

The Capability Maturity Model® Integration℠ (CMMI℠) project was initiated to address these problems. Figure 1 shows how integrating material into the CMMI models adds up as you compare the number of model elements in discipline-specific models to those in CMMI models.

CMMI is sponsored by the U.S. Department of Defense Office of the Under Secretary of Defense, Acquisition, Technology, and Logistics and the Systems Engineering Committee of the National Defense Industrial Association. The CMMI project has been developing a product suite, including CMMI models, appraisal method, and training, since 1998. Version 1.0 (V1.0) was released in August 2000 and Version 1.1 (V1.1) was released in January 2002. During the next few years, V1.1 will remain stable. No updates are planned for the next three years.

The experience gained in launching the Software CMM (SW-CMM®) in the early 1990's led the CMMI project to move fairly quickly from a V1.0 to a V1.1. We felt we had an initial operating capability with V1.0, but needed to commit resources to reach a final operating capability of the baseline models. This approach drove the decision to have a formal public review period early in 2001 to gather initial user input. This was then coupled with a second phase of pilots using the model before publishing updates of the CMMI models and the Standard CMMI℠ Appraisal Method for Process Improvement (SCAMPI℠) method.



We created a Change Control Board (CCB) populated with experts from across government and industry as well as the Software Engineering Institute (SEI). Guidelines were established to limit V1.1 changes to those addressing change requests (CRs) that identified elements that were "broken." This meant that we would not restructure the model, nor add material for new elements beyond those in V1.0. CRs that called for new process areas or combining process areas were deferred because they would result in more change than we wished to see for V1.1.

Probably the most controversial decision made for V1.1 was to maintain two representations, or architectures, of the CMMI model [1]. The systems engineering and software engineering communities that had grown accustomed to two contrasting architectures for their respective discipline-specific models felt it best to maintain both the continuous representation from the Electronic Industries Alliance/Interim Standard (EIA/IS) 731 heritage and the staged representation from the SW-CMM heritage [2]. As the CMMI models have evolved, the model content has remained consistent for both continuous and staged representations.

V1.1 Themes: Stability and Usability

Stability of the CMMI Product Suite refers to maintaining essential content and structure while evolving the products to improve quality. We knew this theme was critical to users considering transition so they would have a relatively unchanging product for building their transition plans [3]. We knew that early users of the CMMI Product Suite would uncover some need for clarifications and improvements [4]. However, we gained confidence from comments during later pilot tests that the V1.0 material was, in fact, ready for operational use. Therefore, to promote transition and to preserve the investment of early adopters, the decision was made to focus V1.1 changes primarily on corrections of errors and essential clarifications that were designed to avoid confusion. The multi-organizational CCB has helped maintain an approach that honors requests for clarification with minimal impact to the "required" and "expected" elements of CMMI models.

CMMI stability could have been fully preserved by making no changes to the product suite after the CMMI V1.0 release, but usability considerations dictated otherwise. The CRs confirmed what we suspected – a group of authors from various backgrounds and environments writing documents for users with various backgrounds and environments used words and phrases that sometimes needed to be clarified. Without that clarification, process improvement groups and appraisal teams may spend valuable time on inter-

® Capability Maturity Model and CMM are registered in the U.S. Patent and Trademark Office.
℠ CMMI, CMM Integration, SCAMPI, SCAMPI Lead Assessor, and SCE are service marks of Carnegie Mellon University.

pretation that would clearly be better spent on improvement or its validation. We found that every process area had some room for such improvement. Some process areas such as Process and Product Quality Assurance had relatively few deficiencies. Others, particularly those that reflected more cross-discipline integration, showed a greater need for explanatory statements. Clarifications have improved the usability of the model without adversely affecting its stability.

Model Changes Equal Greater Clarity

The initial CR reviews brought confidence that most corrections could be made in informative material, rather than across the required and expected elements of the model. In the end, only one goal was deleted by a restructuring of the Organizational Process Definition process area. Three other goals have been changed or clarified: one each in the Organizational Training, Technical Solution, and Product Integration process areas. At the practice level, about 10 percent had wording clarifications, and two practices were deleted.

The V1.1 models have retained the same process areas. Further, these process areas belong to the same process area categories as V1.0 of the continuous representation, and to the same maturity levels as V1.0 of the staged representation. Figure 2 shows a high-level summary of V1.1 process areas and their position in each representation.

Most of the changes have focused on assuring that key terms are used consistently throughout all the models. "Process capability" is a typical example. The term has a specific meaning for experts in statistical process control, but it often seemed appropriate to describe improvements in the capability dimension of the continuous representation. In V1.1, only the statistical meaning is used. Similar clarifications were provided for terms such as "life cycle," "process," and "process area."

Relationships across the engineering process areas were clarified. Of note was the elimination of the overlap between "make-or-buy" analysis in the Technical Solution and Supplier Agreement Management process areas. Greater attention to architectural analysis and design was requested in CRs, and an enhanced specific practice addressed this area.

Generic practice (GP) content was strengthened for V1.1. Improved elaborations are now provided for "Plan the Process" (GP 2.2), "Provide Resources" (GP 2.3), and "Monitor and Control the Process" (GP 2.8). Model authors also clarified the relationship of several of the GPs to their associated process areas based on feedback from some of the pilot appraisals using the continuous representation.

Release	PAs/ FAs		Goals/ Themes*		Activities/ Practices**	
SW-CMM V1.1	18		52		316	
SW-CMM V2C	19	61	62	199	318	1566
EIA/IS 731	19		77		383	
IPD-CMM V0.98	23		60		865	
CMMI V1.0 SE/SW	22		70		417	
CMMI V1.0 SE/SW/IPPD	24		76		460	
CMMI V1.1 SE/SW/IPPD	24		75		458	

* Ratable components
** Key to implementation effort

Figure 1: *Model Measures*

Some terms were replaced with more appropriate substitutes. "Capture" was replaced with "document" or "record." "Process capability model" was replaced, where appropriate, with "process performance model." And, as will be evident in the following section, we replaced the term "assessment" with "appraisal" as appropriate.

Appraisal Changes: A Shift in Approach

One of the significant additions to CMMI V1.1 is a single, common method for external evaluations as well as internal appraisals for process improvement. Because of the strong correlation of the Software Capability Evaluation (SCE[SM]) V3.0 with the familiar CMM-Based Appraisal for Internal Process Improvement, the SCE was chosen as a source document for the new CMMI Method Description Document. A separate implementation guide for source selection and contract monitoring will also be provided to describe the unique characteristics of an appraisal when this kind of external use is planned.

The other key effort that the appraisal team undertook was to apply "lessons learned" from the community. Two key concepts are now part of the V1.1 release. First, the idea of "triage" has been more explicitly described. This description has the appraisal team focusing more of their attention on the areas of greater uncertainty and moving quickly through the process areas where clear evidence has been delivered. Second, when conducting an appraisal, the appraisal team will clearly expect that the appraised organization has done its homework to prepare for the visit. The materials provided by the appraised organization will be used from the beginning of the appraisal, so that the appraisal team's effort shifts from a "discovery" approach to a "verify and validate" approach. Pilot use of these approaches has shown marked improvement in the time required to assure goal satisfaction.

Figure 2: *CMMI Process Areas*

CONTINUOUS REPRESENTATION

Category	Process Areas
Process Management	Organizational Process Focus
	Organizational Process Definition
	Organizational Training
	Organizational Process Performance
	Organizational Innovation and Deployment
Project Management	Project Planning
	Project Monitoring and Control
	Supplier Agreement Management
	Integrated Project Management
	Risk Management
	Quantitative Project Management
	Integrated Teaming (IPPD)
Engineering	Requirements Management
	Requirements Development
	Technical Solution
	Product Integration
	Verification
	Validation
Support	Configuration Management
	Process and Product Quality Assurance
	Measurement and Analysis
	Causal Analysis and Resolution
	Decision Analysis and Resolution
	Organizational Environment for Integration (IPPD)

STAGED REPRESENTATION

Level	Focus	Process Areas
5 Optimizing	Continuous Process Improvement	Organizational Innovation and Deployment
		Causal Analysis and Resolution
4 Quantitatively Managed	Quantitative Management	Organizational Process Performance
		Quantitative Project Management
3 Defined	Process Standardization	Requirements Development
		Technical Solution
		Product Integration
		Verification
		Validation
		Organizational Process Focus
		Organizational Process Definition
		Organizational Training
		Integrated Project Management
		Risk Management
		Decision Analysis and Resolution
		Organizational Environment for Integration (IPPD)
		Integrated Teaming (IPPD)
2 Managed	Basic Project Management	Requirements Management
		Project Planning
		Project Monitoring and Control
		Supplier Agreement Management
		Measurement and Analysis
		Process and Product Quality Assurance
		Configuration Management
1 Initial		

Version 1.1 Training

Because most of the changes to the model approved for V1.1 are in the informative material, few changes will be required to the training slides for CMMI courses [5]. More significant is the work required to assure that all of the instructors of the revised material understand the clarifications that were made so that they can effectively teach the revised material. We do not envision any need to provide "delta" training to those who have taken the V1.0 courses. Those who have attended V1.0 courses may wish to print a copy of the model version comparison document we have provided. The Word document with redlines can be found in the "CMMI Models" section of the CMMI Web site <www.sei.cmu.edu/cmmi/>.

Discoveries from the Pilots

In October 2000, a call for participation was issued for the Phase II Pilot Program to gather experiences using V1.0. Eight pilot tests were conducted. The appraisal method used was SCAMPI V1.0 with some experimental variations intended to improve the efficiency of the method. Phase II pilots included observers from the CMMI project who collected measurements to gauge progress against the defined appraisal goal of 100 hours. Observers used this goal, 100 hours or 5.7 hours per process area, for the on-site period of a SCAMPI appraisal using the CMMI model for Systems Engineering and Software Engineering (CMMI-SE/SW), with the 18 process areas that define Target Profile 3 or, equivalently, Maturity Level 3.

Across the eight pilot appraisals, an average of 5.3 hours per process area during the on-site period was achieved against the objective of 5.7 hours per process area. In particular, SCAMPI pilots using improved data gathering with validating interviews achieved 5.0 hours per process area. This method, involving additional pre-on-site review of objective evidence, has been included as an improved technique in SCAMPI V1.1.

Other pilot test results reinforced previously known best practices for appraisal conduct. In particular, the importance of appraisal preparation, model training, and in-depth model comprehension by the appraisal team cannot be overemphasized. Given the newness of the CMMI models and appraisal method, no one had vast amounts of experience with their use. However, those with a working knowledge attained from prior appraisals or transition experience (e.g., mapping of the model to command media) were more adept at applying the model during a SCAMPI.

Sunset Legacy Models

The CMMI Product Suite is intended to gradually replace the SW-CMM and EIA/IS-731. The approaches being taken to sunset these two models are consistent.

The SEI plans no further updates to the SW-CMM model, appraisal methods, and training. After December 2003, the SEI will no longer provide public offerings of "Introduction to SW-CMM" training, although the SEI and SW-CMM transition partners may continue to deliver the training to select organizations. Also after December 2003, SW-CMM appraiser and evaluator training will no longer be offered. Therefore, active SW-CMM appraisers and evaluators will need to transition to CMMI SCAMPI appraiser status by December 2005. SCAMPI will then be the appraisal method of choice.

The Government Electronic Industry Association G-47 committee, the originator of EIA/IS 731, has taken a similar approach. No further updates are planned for this interim standard. The Interim Status has been renewed to cover the transition period while the CMMI Product Suite is being adopted, but no further extensions are currently envisioned.

Summary

The year 2001 has been a remarkable year for the CMMI Product Suite. The improvement to the product suite from reviews and rewrites will be obvious to the using community as it begins reading and using CMMI materials. Now we look forward to working with you to ease adoption of this even better model that builds upon and integrates the firm process improvement framework of the legacy models.

References

1. Shrum, S. "Choosing a CMMI Model Representation," <www.stsc.hill.af.mil/crosstalk/2000/jul/shrum.asp>, Oct. 2001.
2. SEI. Capability Maturity Model® (SW-CMM®) for Software, <www.sei.cmu.edu/cmm>, Sept. 2001a.
3. SEI. CMMI℠ Publications and Transition Materials, <www.sei.cmu.edu/cmmi/publications/pubs.html>, Sept. 2001f.
4. SEI. CMMI℠ Product Suite, <www.sei.cmu.edu/cmmi/products/products.html>, Sept. 2001e.
5. SEI. Education and Training Courses, <www.sei.cmu.edu/products/courses/courses.html>, Sept. 2001d.

Additional Reading

1. SEI. Systems Engineering Capability Maturity Model®, <www.sei.cmu.edu/cmm/se-cmm.html>, Sept. 2001b.
2. SEI. CMMI℠ Tutorial, <www.sei.cmu.edu/cmmi/publications/stc.presentations/tutorial.html>, Sept. 2001c.
3. SEI. Software Engineering Information Repository, <http://seir.sei.cmu.edu>, Sept. 2001g.
4. SEI. Technology Adoption, <www.sei.cmu.edu/adopting/adopting.html>, Oct. 2001h.
5. SEI. "Bibliography on Integrated Product and Process Development," <www.sei.cmu.edu/cmmi/publications/ippd-biblio.html>, Nov. 2001i.

About the Author

Mike Phillips is director of Special Projects at the Software Engineering Institute (SEI), a position created to lead the Capability Maturity Model® Integration℠ project for the SEI and the steering group. He was previously responsible for transition enabling activities at the SEI. Prior to his retirement as a colonel from the Air Force, he managed the $36 billion development program for the B-2 in the B-2 Special Programs Office and commanded the 4950th Test Wing at Wright-Patterson AFB, Ohio. Phillips has a bachelor's degree from the Air Force Academy, and master's degrees from Georgia Tech, from the University of Southern California, and from Salve Regina College and the Naval War College.

4500 Fifth Ave.
Pittsburgh, PA 15213
Phone: (412) 268-5884
Fax: (412) 268-5758
E-mail: dmp@sei.cmu.edu

Publisher's Note

CROSSTALK's parent organization, the Software Technology Support Center (STSC), is a technology transition partner with the Software Engineering Institute (SEI). Organizations new to software process improvement or to the CMMI can receive additional help with understanding the information from the SEI and their technology transition partners at the STSC's and other Web sites listed on page 27.

CMMI Appraisal Methodologies: Choosing What Is Right for You

Ilene Minnich
SECAT LLC

Rumors abound about the horrors of Capability Maturity Model® Integration℠ (CMMI℠) appraisals: "Two weeks straight! Twenty-one hour days!! And we never even got done!!!" Yes, I lived through some of those early days, and I had to wonder: How are we going to sell the benefits of CMMI appraisals to the world? What organization would willingly subject itself to that kind of pain in the name of internal process improvement? And how many fools – I mean consultants – would hang out their shingle as a CMMI lead appraiser to provide the supporting infrastructure that is needed for widespread adoption of the CMMI? A year later I feel confident that CMMI appraisals will find a place in every organization that is serious about process improvement. The keys to success are education, preparation, and pre-work.

Starting with the Appraisal Requirements for Capability Maturity Model® (CMM®) Integration℠ (CMMI℠) (ARC) Version 1.0 [1], the authors of the CMMI product suite laid out the requirements for three classes of appraisal methods. This is important because it recognizes that an organization can get benefits from internal appraisals at various levels of resource expenditures.

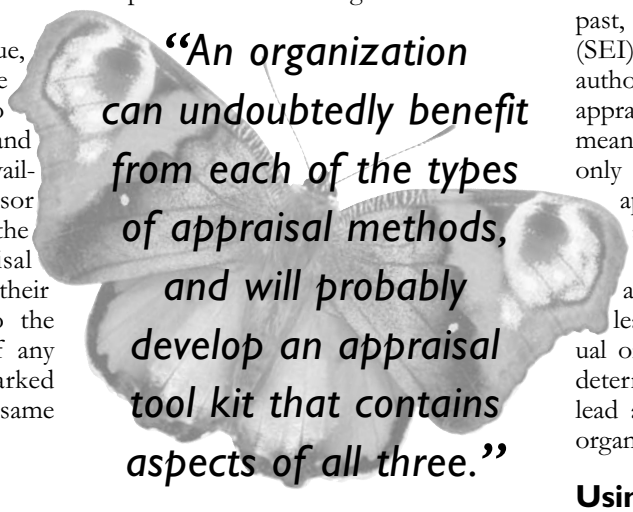
Of course this has always been true, but the ARC formalizes the three classes by mapping requirements to them, which provides a consistency and standardization that has not been available with any of the CMMI predecessor models. It also allows organizations the freedom to develop an appraisal methodology that works best for their organization, and once mapped to the ARC appraisal classes, the results of any appraisal can be easily benchmarked against other appraisals from the same class.

Comparing the Different Appraisal Classes

The characteristics of the CMMI appraisal classes are summarized in Table 1 (see page 8). Class A describes a full appraisal, usually performed by a team of six to 10 people, primarily drawn from inside the organization being appraised. A class A appraisal is expected to be the most accurate, designed to maximize buy-in from the appraisal participants, and leaves the organization with the best understanding of issues that need to be fixed and strengths that should be shared. The Standard CMMI Appraisal Method for Process Improvement (SCAMPI) describes a class A appraisal method [2].

Class B describes a smaller scale appraisal methodology, sometimes called a mini-appraisal or a pre-appraisal. A class B appraisal can be accomplished with a smaller team of expert appraisers over a reduced number of days. It can be used as a substitute for a full appraisal or to spot-check the organization between full appraisals.

Class C describes the least intensive appraisal methodology, sometimes called a micro-appraisal or questionnaire-based appraisal. A class C appraisal can be used to get a rough idea of the current state of the practice within an organization.



The length of time needed to complete an appraisal can be significantly more for a class A appraisal than for a class B or C appraisal. Many factors contribute to the time needed to complete an appraisal. Some examples are the size and complexity of the organization, the number of process areas and capability levels or maturity levels covered, the size of the appraisal team, the training and experience level of the appraisal team, and the amount and rigor of evidence review.

The goal for completing the on-site portion of a full class A or SCAMPI appraisal is roughly two weeks. The majority of this time will be spent gathering and evaluating evidence to determine appropriate coverage of the model's practices. A rough estimate for a class B appraisal is one week. This may be accomplished through less rigorous evidence collection and review, perhaps by relying more on interviews or "spot checking" for practice compliance. Class C appraisals may be only hours long and are likely to be based on questions and answers, with little examination of evidence.

Another difference between appraisal classes is the expected training or experience level of the lead appraiser. As with the Software CMM lead appraisers in the past, the Software Engineering Institute (SEI) is responsible for training and authorizing lead appraisers for SCAMPI appraisals for the CMMI. This does not mean that others can not lead an appraisal, only that if you want or need "official" appraisal results you must use a SEI authorized SCAMPI lead appraiser.

There are no plans currently to authorize appraisers for class B or C lead appraisers. It is up to the individual organization planning an appraisal to determine the qualifications needed by the lead appraiser to meet the needs of the organization's appraisal.

Using an Appraisal Tool Kit

An organization can undoubtedly benefit from each of the types of appraisal methods, and will probably develop an appraisal tool kit that contains aspects of all three. For example, an organization may develop the following:

- A questionnaire or checklist (class C) to be used quarterly to "remind" everyone of the processes that should be followed.
- A mini-appraisal (class B) that will be performed internally every year to determine the current state of the practice.
- A full appraisal (class A) that will be performed by an outside source every two to three years or as needed for contract procurement.

The combination of these three classes will allow each organization to customize its appraisals to best meet its process improvement needs.

Characteristics	Class A	Class B	Class C
Usage Mode	1. Rigorous and in-depth investigation of process(es). 2. Basis for improvement activities.	1. Initial (first time). 2. Incremental (partial). 3. Self appraisal.	1. Quick-look. 2. Incremental.
Advantages	Thorough coverage; strengths and weaknesses for each PA investigated; robustness of method with consistent, repeatable results; provides objective view; option of ISO 15504 conformance.	Organization gains insight into own capability; provides a starting point, or focuses on areas that need most attention; promotes buy-in.	Inexpensive, short duration, rapid feedback.
Disadvantages	Demands significant resources.	Does not emphasize depth of coverage and rigor, and cannot be used for level rating.	Provides less buy-in and ownership of results, not enough depth to fine tune process improvement plans.
Sponsor	Senior manager of organizational unit.	Any manager sponsoring a SPI program.	Any internal manager.
Team Size	4-10 people plus an appraisal team leader.	1-6 people plus an appraisal team leader.	1-2 people plus an appraisal team leader.
Team Qualification	Experienced.	Moderately experienced.	Moderately experienced.
Appraisal Team Leader Requirements	Lead appraiser.	Lead appraiser or person experienced in method.	Person trained in method.
Team Composition	External and internal.	External or internal.	External or internal.

Table 1: *Characteristics of CMMI Appraisal Classes* [1]

Reducing Appraisal Pain

Preparation for the appraisal always plays a big part in its success. As with all of the predecessors to CMMI, the definition of the scope of the organization is probably the one decision that most affects the time to complete the appraisal itself. Since CMMI can be used to evaluate the activities associated with systems engineering, software engineering, integrated product and process development (IPPD), and acquisition, be aware that the broader the scope of the organization the more people will be involved in the scope of the appraisal. A broader organization, or an organization now looking at including more "disciplines," takes more time to appraise. This has always been true, but many appraisal sponsors may not be aware of the obvious correlation.

Another way to lessen the pain of an appraisal is to shift as much work as possible away from the "on-site" portion of the appraisal and complete it beforehand. There are many variations of this. Suggestions for appraisal pre-work include mapping the organization's processes to CMMI, gathering and/or reviewing evidence, distributing and completing CMMI-based questionnaires, and developing interview questions for use during the appraisal. The better the data your appraisal team starts with, the less time it will take the team to complete.

Probably the biggest contributor to the success of your CMMI appraisal will be in providing your appraisal participants with the proper level of education, especially if they have some preconceived notions based on the use of predecessor models. The appraisal sponsors need to have realistic expectations concerning the scope of the organization, the number of appraisal participants and their areas of expertise, and the use of each of the appraisal classes. The appraisal team and the supporting staff responsible for the appraisal pre-work need to understand the CMMI, the requirements and methodology for the appropriate class of appraisal, and how to map or translate the work being performed in the organization to the CMMI. The remainder of the appraisal participants most likely will not need any special CMMI training.

Conclusion

Thorough planning and pre-work of a CMMI appraisal may be more important than ever before, especially if your organization is planning to broaden its definition of organization or include additional disciplines and activities. Setting expectations, educating participants, and mapping terminology are key to the success of an appraisal. Developing an appraisal tool kit, including the different appraisal classes will allow your organization to meet its process improvement needs in an efficient and effective manner.◆

References

Note: All these documents are available on the Web at <www.sei.cmu.edu>.

1. CMMI Product Development Team. Appraisal Requirements for CMMI (ARC), Version 1.0. CMU/SEI-2000-TR-011, Aug. 2000.
2. Standard CMMI Appraisal Method for Process Improvement: Method Description, Version 1.0. CMU/SEI-2000-TR-009, Oct. 2000.
3. CMMI Product Development Team, Capability Maturity Model Integration for Systems Engineering/Software Engineering/Integrated Product and Process Development, Version 1.1. Continuous Representation. CMU/SEI-2000-TR-019, Aug. 2000.

About the Author

Ilene Minnich is a SECAT LLC founding partner and has worked as a process improvement and appraisal consultant to more than 70 organizations worldwide. She is a part-time visiting scientist at the Software Engineering Institute (SEI) and works with the Capability Maturity Model® (CMM®) Integration℠ training team. She is an experienced CMM author, appraiser, and trainer. Previously at Hughes Aircraft Company, Minnich was responsible for the Software-CMM deployment and conducting Systems Engineering-CMM appraisals at all company-wide systems engineering organizations. Minnich has a bachelor's of science degree from Lehigh University and a master's degree from the University of Southern California.

SECAT LLC
14742 Beach Blvd. #405
La Mirada, CA 90638
Phone: (714) 449-0423
Fax: (714) 449-0256
E-mail: iminnich@secat.com

To CMMI or Not to CMMI: Issues to Think About

Winifred Menezes
Consultant

Version 1.02 of the Capability Maturity Model® Integration℠ (CMMI℠)-Systems Engineering/Software Engineering and CMMI-SE/SW/Integrated Product and Process Development were released more than a year ago. At the time this article was written, a cleaner and more stable version 1.1 was due to be released in January 2002. A number of organizations have decided to adopt the new model as a guide for their process improvement program (see <www.sei.cmu.edu/cmmi/publications/early-adopters.html> for a list of early adopters). Others are asking questions like: Should we adopt the CMMI? When is the right time to transition? Which model is suitable for our business? Which representation makes sense? This article describes a number of scenarios and discusses the pertinent issues for each. But first, it begins with some general information about the CMMI.

The Capability Maturity Model® (CMM®) Integration℠ (CMMI℠) in its present form is a collection of best practices for the "development and maintenance" of both "products and services." The model was developed by integrating practices from four different CMMs – the "source models:" the CMM for software, for systems engineering, for integrated product development (IPD), and for acquisition.

Organizations can use the model as a guide for improving their ability to develop (or maintain) products (and services) on time, within budget, and with desired quality. During the past decade many organizations have used CMM and CMM-like concepts to bring order to their software development processes. The CMMI allows these organizations to continue focusing only on the discipline of software.

Additionally, it also provides these organizations the framework for enlarging the focus of process improvement to other areas that also effect product development – the discipline of systems engineering. During the past decade, new and effective concepts for organizing developmental work have surfaced and been adopted such as concurrent engineering or the use of integrated teams. Organizations using (or wishing to adopt these ideas) can also find support in the CMMI by using the model with integrated product and process development (IPPD) additions.

Finally, organizations that acquire components or services as a substantial part of development will find the acquisition additions useful. (CMMI-Systems Engineering (SE)/Software Engineering (SW)/IPPD/Acquisition (A) Version 1.02d draft is available for review and piloting.) CMMI-SE/SW, CMMI-SE/SW/IPPD as well as CMMI-SE/SW/IPPD/A are available at the Software Engineering Institute's (SEI) Web site <www.sei.cmu.edu/cmmi>.

Representations

The CMMI has yet another complexity to it: the representations, staged and continuous. Philosophically there are two different approaches to process improvement. One focuses on the organization as a whole and provides a road map of successive stages aimed at improving the organization's ability to understand and control its processes. This approach is the basis for the staged representation.

"During the past decade many organizations have used CMM and CMM-like concepts to bring order to their software development processes."

The other approach focuses on individual processes, allowing the organization to choose which process or a set of processes need to have more capability. This is the approach of the continuous representation.

In theory the choice of processes is unconstrained, but in reality increasing the capability of a particular process necessitates that other processes have certain capabilities. So the continuous representation provides a few more routes on the process improvement map. We are talking of two representations – two different views of the same content. The rules for converting one representation into the other have been defined. So a choice of one representation does not preclude the use of another at a later time.

Scenario 1: You Market a Product That Contains Software

Your organization develops and markets a product or a product component that contains software, for example a cell phone or the breaking system for automobiles. During the past decade, you realized that software was a key enabling part of your product; you have been using the Software CMM (SW-CMM) for some years now. In fact some units have been appraised at CMM Level 3, and say they have managed to reduce a good deal of rework.

Should you transition to the CMMI? SEI, who is the custodian of both the SW-CMM and the CMMI, says it will not support the SW-CMM after the year 2003. This pronouncement does not mean the SW-CMM will disappear, but the infrastructure that supports its use (e.g. training, authorization of assessors) will definitely be weakened.

However, there is another more compelling reason to transition to the CMMI. One of the source models that the CMMI was based on was the SW-CMM Version 2 Draft C. This version of the SW-CMM was an improvement on Version 1.1, which is what you and most other people are using. So the CMMI encompasses the experience and lessons learned from the previous 10 or so years of SW-CMM use.

Your Product Life Cycle

When would be a good time to transition? The answer to this depends on your current experiences with the SW-CMM and process improvement, as well as your plans for the future. Are your software

development groups constrained by budget, cost, and product decisions over which they have no influence? Is there a need to bring some structure into the total product-development life cycle? Is senior management planning to use integrated product teams (IPTs) for future development?

If your answer is yes to any of these, then you should begin to introduce CMMI now. You will need to plan for two different types of introduction. The first part of your transition plan is introducing process improvement concepts to the systems engineers, product management, customer representatives or similar groups. Expect the same amount of resistance that the software engineers once had, e.g., "My work is creative, it's different – I can't follow a process." The tools and techniques you initially used with the software groups should be useful.

The second part of your transition plan should concern those using the SW-CMM. If they are well on their way to the next maturity level, it might be better to make them aware of the CMMI. However, let them achieve the maturity level they were aiming for before working with the details of the CMMI. If their process improvement efforts are languishing, then maybe the CMMI will function as a catalyst.

In both cases you will need to think about the "new" process areas in the CMMI. Consider "measurement and analysis." Groups that have achieved CMM Level 3 or higher in the SW-CMM will already have some of the practices in place for this process area. Other new process areas, for example those in the engineering category, might require more effort. On the other hand, you may be pleasantly surprised to find that the software groups had these in place already. The objective of this two-pronged introduction should be that all relevant groups ultimately have the same level of process capability.

Your Process Improvement Experience

Your systems engineering groups may already have the same amount of process awareness as your software groups. You may be one of the organizations that has tailored out the word "software" and applied the SW-CMM to non-software development too. In this case, the choice of when to transition should depend on the current effort of process improvement.

If groups are working towards a maturity level, make them aware of the new model but wait for them to reach their maturity objective. If improvement work has reached a standstill, then CMMI may be the refocus point. However, first find out why improvement work is at a standstill before attempting to rally people around a slightly different flag pole.

Which model should be used? If you use or plan to use IPTs, then choose the CMMI-SE/SW/IPPD. If not, stay with the CMMI-SE/SW.

Finally, which representation should be used? Since you have experience with the SW-CMM, staying with the staged representation would be the easiest. If, however, your organization has fallen into a "level hunt" (levels for the sake of levels), then you may want to break the circle by using the continuous representation. Some organizations with a good understanding of processes and process improvement prefer the continuous representation because it provides more granularity and flexibility. Remember there are equivalency rules between the two representations, so you can get the benefit of both worlds.

Suppose, however, you are one of those organizations that has not used the SW-CMM. In this case the issue is not transition but adoption. If you want to start improving your processes, start with the CMMI by implementing the practices on all development groups within the limits of your improvement budget. Since you are new to the improvement game, you will find better guidance in the staged representation. So unless there is some compelling reason, choose the staged representation.

Scenario 2: You Develop Only the Software Component

In this scenario, you are a software development unit within a larger enterprise. That is, other units develop requirements, some of which will be met by the software your group develops. Yet other units take what you deliver and integrate it with other components into a product or service.

You are interested in applying the CMMI to software only. This still means that the whole of CMMI-SE/SW (or CMMI-SE/SW/IPPD in the event you use or plan to use IPTs) is applicable. The process area descriptions contain amplifications for systems engineering, software engineering, and IPPD. The amplifications contain more information about how a practice could be applied within a particular discipline. You could skip the amplifications for systems engineering (and IPPD), but the process area would still be applicable.

The Best Time to Transition

When should you start using the CMMI? Just as in Scenario 1, that depends on whether you are transitioning from one of the source models or adopting a process improvement model for the first time. Similarly, the discussion from Scenario 1 regarding which model and which representation are applicable applies for this scenario, too.

The issues you will particularly need to think about are "use of" or "interpretations of" some of the engineering process areas, such as requirements development, product integration, and validation. Since other units in your enterprise are responsible for developing requirements, integrating the components, and validating that the product meets the customer needs, your unit will need to study these process areas and decide if there is a useful mapping between their practices and the scope of your unit's responsibilities.

Scenario 3: Software Is Your Product

This scenario is a combination of Scenarios 1 and 2 with a twist. Your organization develops and markets a software product such as a word processor, financial system, networking software, or a game. You have most certainly heard about the SW-CMM. You may be on the verge of using the model and are now wondering about introducing a model that will be phased out after a few years. Or, you may have started using it either informally as a source of best practices or more formal as the basis for a sponsored and planned process improvement program.

Most of the discussion in Scenario 1 about transitioning or adopting the CMMI, as well as which representation to use, is applicable to you. Since you market your software product, you should also consider the "systems" part of the software product development. This includes issues like how the software will be used and how it will be marketed and delivered. The combination of management and engineering process areas would be a good guide in your process improvement work.

Scenario 4: Your Software Process Improvement Is Based on ISO 15504

This scenario, really a variant of both Scenario 1 and 2, is that you have experience using ISO 15504[1] (also known as SPICE) as your guide to software process

improvement. The reference model in ISO 15504 covers the software life cycle, so any need to enlarge the scope of your process improvement to all product development would be one reason to move to CMMI. ISO 15504 has a continuous architecture, so you would probably find the continuous representation easier to use.

Another reason to adopt the CMMI is that the revised ISO 15504 will no longer have a reference model.

The standard will have guidance for performing appraisals as well as compliance requirements for suitable reference models. Both the CMMI model as well as the appraisal method released by the SEI are ISO 15504-compliant. This means that if you were to be appraised by a rigorous appraisal process (like the one released by the SEI) you would fulfill any ISO-15504 requirements you may have.

Scenario 5: Your Product Does Not Have Software.

The product or service you develop and market contains no software. Do you need to think about CMMI? The answer is yes; the CMMI applies to all product (or service) development.

Since you do not "do" software, you have obviously not bothered with the SW-CMM. But you may have used Electronic Industries Alliance (EIA) 731 (or one of its predecessor models, Systems Engineering Capability Appraisal Model or SE-CMM). Since EIA 731 is an interim standard and was one of the source models for the CMMI, there is reason to transition to the CMMI.

The continuous representation will be most suitable since that is what you are used to from EIA 731. You would not be interested in the software amplifications within the process areas, but the systems engineering and perhaps IPPD amplifications would be of use. So you would choose CMMI-SE/SW or CMMI-SE/SW/IPPD.

Wait until you reach a milestone in the current process improvement efforts before transitioning to CMMI.

Conclusion

Every organization's particular situation is unique. In all probability none of the scenarios above will exactly fit your organization. However, this general discussion should give you some idea of what issues and problems you will face. The answers to the following set of questions are ultimately what should guide an organization's CMMI adoption strategy. Possible answers to these questions were the basis of the five scenarios above. Naturally your unique answers should guide your CMMI adoption strategy:

- What are the organization's business goals?
- What product/service does the organization develop/maintain?
- What is the product life cycle and development/maintenance organization?
- How much process improvement experience do the various units within the organization have?
- Would it make sense to enlarge the current process improvement effort to other parts of the development organization?
- When will the organization meet the next improvement milestone?

Good Luck!◆

References

1. CMMI for Systems Engineering/Software Engineering, Version 1.02 (CMMI-SE/SW, Version 1.02) Continuous Representation CMU/ SEI-2000-TR-029, ESC-TR-2000-094, CMMI Product Development Team <www.sei.cmu.edu/cmmi>.
2. CMMI for Systems Engineering/Software Engineering, Version 1.02 (CMMI-SE/SW, Version 1.02) Continuous Representation CMU/SEI-2000-TR-028, ESC-TR-2000-093, CMMI Product Development Team <www.sei.cmu.edu/cmmi>.
3. CMMI for Systems Engineering/Software Engineering/Integrated Product and Process Development, Version 1.02 (CMMI-SE/SW/IPPD, Version 1.02) Continuous Representation CMU/SEI-2000-TR-031, ESC-TR-2000-096, CMMI Product Development Team <www.sei. cmu. edu/cmmi>.
4. CMMI for Systems Engineering/Software Engineering/Integrated Product and Process Development, Version 1.02 (CMMI-SE/SW/IPPD, Version 1.02) Staged Representation CMU/SEI-2000-TR-030, ESC-TR-2000-095, CMMI Product Development Team <www.sei. cmu.edu/ cmmi>.
5. CMMI for Systems Engineering/Software Engineering/Integrated Product and Process Development/Acquisition, Version 1.02d DRAFT (CMMI-SE/SW/IPPD, Version 1.02d DRAFT) Continuous Representation CMU/SEI-20XX-TR-XXX, ESC-TR-20XX-XXX, CMMI Product Development Team <www.sei. cmu. edu/cmmi>.
6. CMMI for Systems Engineering/Software Engineering/Integrated Product and Process Development/Acquisition, Version 1.02d DRAFT (CMMI-SE/SW/IPPD, Version 1.02d DRAFT) Staged Representation CMU/SEI-20XX-TR-XXX, ESC-TR-20XX-XXX, CMMI Product Development Team <www.sei. cmu.edu/cmmi>.
7. International Organization for Standardization and International Electrotechnical Commission. Information Technology: Software Process Appraisal. Geneva, Switzerland: 1998.

Note

1. ISO 15504, a technical report published by the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC), is expected to become a standard some time in the future. ISO 15504 consists of nine parts, covering the method to be used for assessing the software development processes. For an objective appraisal (or evaluation) processes are compared against a particular model or standard, called the reference model. The SW-CMM and CMMI are reference models as is part two of ISO 15504.

About the Author

Winifred Menezes has more than 20 years experience in software engineering, software process improvement, and training. She has lived and worked in the United States since 1997. Previously, Winifred worked as a consultant at Cap Gemini and later at ABB Corporate Research, both in Sweden. At ABB she was instrumental in starting ABB Sweden's software process improvement program. Winifred is a member of the Capability Maturity Model® Integration℠ (CMMI℠) product team. She is also an authorized instructor of the "Introduction to Software-CMM" and the "Introduction to CMMI." In September 2000 she received a "certificate of appreciation" for her contribution to the CMMI product suite. She received a master's of science in mathematics from the University of Islamabad, Pakistan and a Fil Kand in computer sciences from the University of Upsala, Sweden.

3406 Bitterwood Place, J202
Laurel, MD 20724
Phone: (301) 317-1425
E-mail: winifred.menezes@ieee.com

Transitioning From SA-CMM to CMMI in the Special Operations Forces Systems Program Office

Donald R. Michels
Warner Robins Air Logistics Center

Bonnie Bollinger
Consultant

In 1997, the Special Operations Forces Systems Program Office (SOF SPO) at Warner Robins Air Logistics Center, Robins Air Force Base, Ga., also known as the Special Operations Forces System Program Office Directorate or the LU Directorate, began a partnership with the Software Engineering Institute (SEI), Carnegie Mellon University, on the use of SEI's Capability Maturity Model® (CMM®). This partnership was started as a result of a desire to implement continuous process improvement as an institutionalized way of doing business within the directorate. Our initial model was the Software Acquisition-CMM®. We continued to use this model until late fall 2000. At that time, the directorate converted to the CMM Integration℠/Systems Engineering/Software Engineering/Integrated Product and Process Development/Acquisition (hereafter referred to as CMMI-A). This article explains why the decision was made to change to the CMMI-A, how we became a pilot organization to test the validity of the model, our training on the model, and what we learned during the course of conducting a pilot appraisal.

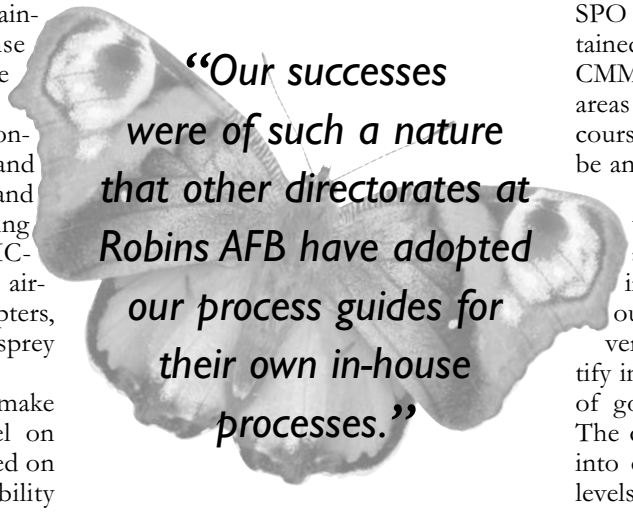
The Special Operations Forces Systems Program Office (SOF SPO) at Warner Robins Air Logistics Center (WR-ALC), Robins Air Force Base, Ga., provides combat weapon systems, equipment, and agile combat support for special operations and Air Force helicopter forces. We deliver best value sustainment and contingency response through world-class cradle-to-grave leadership and management.

The SOF SPO has primary responsibility for systems engineering and technical services to the SOF Fleets and Combat Search and Rescue, consisting of AC-130H/U gunships, MC-130E/H/P Combat Talon/Shadow aircraft, MH-53 J/M Pave Low helicopters, H-1/H-60 helicopters, and CV-22 Osprey tilt-rotor aircraft.

We recognized that in order to make improvements, we needed a model on which to pattern processes. We settled on the Software Acquisition-Capability Maturity Model® (CMM®) (SA-CMM) as a model that would provide the ability to assess our processes. The SA-CMM was structured, contained specific goals, established levels of competence, and provided the framework needed to facilitate improvement. Using the SA-CMM, the SOF SPO established an improvement infrastructure, developed a life-cycle checklist of its directorate's processes, and started improvement efforts.

The SOF SPO improvement program, known as the LU Acquisition and Sustainment Process Improvement/Reengineering Effort or ASPIRE, made many improvements during the next three years. Our successes were of such a nature that other directorates at Robins AFB have adopted our process guides for their own in-house processes. There was, however, one tiny barrier to complete satisfaction with the program. It was the use of the SA-CMM as a model for improvement.

Software development and management in relationship to acquisition and sustainment is a very small part of what SOF SPO does. Since the SA-CMM referred to primarily software acquisition and development, it was initially viewed as a turn-off to the LU work-force. They wanted something more related to what they did. In the course of due time, it even dropped all mention of software from its process improvement activities.



The work-force simply did not fit the model, or more accurately, the model did not fit it. So it continued to search for a model on which to base its program, but did not abandon the SA-CMM in the meantime.

In the fall of 2000, Dr. Thomas Christian, chief engineer, and Greg Stanley, deputy director, attended a conference held in Washington, D.C. During the conference, SEI presented, in draft form, a new model: Capability Maturity Model Integration℠/Systems Engineering/Software Engineering/Integrated Product and Process Development/Acquisition (hereafter referred to as CMMI-A). This model dealt with processes within an acquisition organization. Now here was something the SOF SPO could sink its teeth into. It maintained the proven structure of earlier CMM models, but also contained process areas (PAs) on acquisition. There were, of course, several questions that needed to be answered about the model.

When the CMMI-A was unveiled, not only did the model address acquisition but it also came in two versions: staged and continuous. This was different! The staged version provides a framework to identify improvement opportunities and a set of goals to guide process improvement. The continuous model groups processes into categories and designates capability levels for each process.

In other words, in the staged representation, the whole organization is appraised and receives a maturity level based on an appraisal of all of the PAs contained in the staged representation. In the continuous model each PA is appraised and assigned a capability level. There is no overall level assigned to the organization. What is the most significant factor here? The organization examines the continuous representation, selects the PAs that either apply to the organization or are PAs the organization wants to improve upon, and uses only the selected PAs during appraisals.

So here we have the old SA-CMM model and the new CMMI-A model with two different representations – staged and continuous. What do we do?

Dr. Christian and Stanley returned to

Robins to brief me and my staff on the new model. We discussed at length our partnership with SEI, the advantages and disadvantages of the SA-CMM and the CMMI-A, and the LU workforce's desire to find a model more suited to what we do. Many questions and answers were posed: "Has it been tested?" "No;" "How do we know it will work?" "We don't;" and "Has anyone been appraised using the model?" "No."

In the end, we decided that even with these negative responses, the CMMI-A with continuous representation was worth our time and effort as a model for process improvement. Finally, we had a model that fit us; we did not have to try to fit the model.

Transition to CMMI-A

During their Washington, D.C. trip, Dr. Christian and Stanley discussed the new model with Dr. Jack Ferguson, deputy Acquisition Resources and Analysis director for Software Intensive Systems, Office of the Secretary of Defense, regarding adopting and implementing the model. They pointed out that it had not been tested or implemented anywhere else. Dr. Ferguson asked them to consider using LU as a pilot appraisal organization to test the new model. He would secure the necessary funding to conduct the appraisal. We agreed, and dates were established. We were now on the road to the CMMI-A implementation in the SOF SPO.

We developed three immediate objectives concerning implementing the CMMI-A in LU. First, we wanted training on the new model. Second, we needed to determine the specific PAs on which to be appraised. Third, we wanted a balanced team of appraisers.

The first objective, training, was easy enough. We contracted with SEI to provide training for 30 LU personnel in February 2001. Participating in the training were most of the personnel selected to be on the appraisal team. Training covered the two model representations and the process areas within each; however, the three-day session was only able to present the basics of the model. Training feedback primarily concerned the depth of training they were able to present in three days. While the instructors were very professional and knowledgeable, it was very difficult to come to a level of familiarization with the model in only three days. All agreed that training needed to be longer and more in-depth.

We devoted a lot of effort to ensuring the composition of the appraisal team was as balanced as we could make it. Our considerations in putting the appraisal team together centered on three central thoughts. First, we needed people experienced in conducting appraisals and using previous CMM models. Second, we wanted to make sure the team consisted of people experienced in acquisition and sustainment activities. The latter proved invaluable in interpreting the wording of the model into LU activities. Finally, we needed to make sure we had a team composed of personnel from both external and internal sources.

We were able to achieve all three of these objectives. The team was composed of five LU personnel, three other WR-ALC personnel who had appraisal experience, one representative from SEI, and an experienced appraiser from the Software Technology Support Center, Hill Air Force Base, Utah. We felt this personnel combination provided the expertise and objectivity needed to conduct a thorough appraisal. After establishing the team, our lead appraiser began a series of training sessions designed to teach our functional experts how to conduct an appraisal.

In examining the CMMI-A with continuous representation, we needed to decide which PAs to appraise. In order to accomplish our objectives and the need to "test" the model, we settled on 17 different PAs, see Table 1. These included all the PAs we used in our initial appraisal back in 1997 and added several new PAs. The groundwork for conducting the appraisal was now completed.

Conducting the Appraisal

Conducting the appraisal required a great deal of time and effort. The preliminary step was to survey the work-force. The team then gathered all pertinent data and documentation from the "projects" being appraised. The team then briefed us on how they would conduct the appraisal and developed a series of questions to use in the personnel interviews. Interviews were conducted over a series of days and provided the bulk of information used in determining our capability levels. Their final step was to correlate all the available information and assign a capability rating to each PA.

The team interviewed 47 people, reviewed more than 112 documents, and worked 130 hours during two weeks. The team examined every detail of the selected process areas. Using a team decision process, the results for LU were extremely gratifying. Sixteen of the 17 PAs received a capability rating of Level 2 "A Managed Process." One PA, Organizational Process Focus, was rated Level 3 "Defined Process." This was quite a remarkable achievement for an organization that four years earlier had no documented processes at all and had not used the model to develop its technical activities of systems engineering and configuration management.

Process Area
Organizational Process Focus
Requirements Management
Integrated Teaming (IPPD)
Project Planning
Organizational Environment for Integration (IPPD)
Project Monitoring and Control
Integrated Project Management (IPPD)
Risk Management
Technical Solution
Configuration Management
Product Integration
Supplier Selection & Monitoring
Integrated Supplier Management (IPPD)
Requirements Development
Verification
Validation
Organizational Process Definition

Table 1: *Process Areas*

Especially remarkable was the Level 2 rating given three Integrated Product and Process Development (IPPD) areas: Integrated Teaming, Integrated Project Planning, and Integrated Supplier Management. We had not been able to plan and implement procedures and processes for the IPPDs due to the fact they were not fully defined by SEI until shortly before the appraisal. These ratings demonstrated we had intuitively recognized and were already in conformance with the standards established for the IPPDs at the time they were being developed by SEI.

Based on these results, we developed an action plan to address weaknesses in our processes. Our primary focus will be to establish a measurement and analysis program in order to move us to the next capability level for our selected PAs. Our commitment to process improvement has only been strengthened by our conversion to the CMMI-A.

Lessons Learned

Every decision we make, regardless of the subject, teaches us something. It either reinforces our thought process as being correct, or it shows us where we didn't think things through. Converting to the CMMI-A was the correct decision, but we did learn a few things along the way: some about our program, some

Coming Events

February 11-15
Software Management and Applications of Software Measurement Conferences
Anaheim, CA
www.sqe.com/smasm

March 4-8
The 10th International Conference on Practical Software Quality Techniques and
The 4th International Conference on Practical Software Testing Techniques
New Orleans, LA
www.softdim.com/psqt2002south/

March 25-28
Software Test Automation Conference
San Jose, CA
www.sqe.com/testautomation

April 8-10
Secure E-Business Executive Summit
SecurE-Biz.net
Arlington, VA
www.secure-biz.net

April 9-10
Southeastern Software Engineering Conference
Huntsville, AL
www.ndia-tvc.org/SESEC2002/

April 28–May 2
Software Technology Conference 2002
"Forging the Future of Defense Through Technology"
STC
Salt Lake City, UT
www.stc-online.org

May 13-17
Software Testing Analysis and Review (STAREAST 2002)
STAR EAST 2002
Orlando, FL
www.sqe.com/stareast

about the model, and some about the appraisal process.

We learned that training on the model should be spread out and more in-depth. Our personnel on the appraisal team and the work-force could have used additional training, which would have eased our transition. Feedback from members of the appraisal team leads us to believe had they had a better understanding of the model, we would have received higher capability ratings in several more PAs.

We learned that reviewing organizational documentation should begin as soon as possible. We waited until the last minute to send out our survey. It was quite extensive and required time to respond. Unfortunately, our delay caused recipients to feel pressured to respond quickly, so most ignored it. It also gave us very little time to analyze the results.

We also realized the scope of the appraisal was much too great for the amount of time and number of interviews needed to cover the PAs. The team reviewed more than 112 documents. With all the other things the team needed to accomplish, this was too much to review at a moment's notice. We found examination of 17 different PAs too much to accomplish within a two-week period. We recommend you restrict yourselves to the more important PAs (probably around 10) or make a decision to appraise some now, some later. Careful selection of the PAs would result in a more reasonable effort.

Lastly, while the appraisal methodology calls for the appraisers to construct questions that solicit data about your processes without being direct, our interviewees found this to be both frustrating and confusing. If you decide to follow the current interview methodology, you will need to brief the interviewees that you intend to ask general questions and will be looking for specifics based on their answers. Constructing questions concerning "how, why, and where" may be a more efficient method.

Looking Forward

The SOF SPO has a long and productive relationship with process improvement. From senior leadership down to individual members of our Process Action Teams, we have developed an appreciation of making things better. Our association with the CMM has only enhanced this appreciation. The CMMI-A is our guide to the future. We are convinced our involvement with the CMMI-A is a key factor in why we have developed a "passion for excellence."◆

About the Authors

Donald R. Michels, GS-15, is the director of Special Operations Forces Systems Program Office at Warner Robins Air Logistics Center, Robins Air Force Base, Ga. He is the U.S. Air Force's single manager for fleet management of Air Force Special Operations Command fixed-wing aircraft and all U.S. Air Force helicopters. He directs more than 420 personnel at Robins Air Force Base, Wright-Patterson Air Force Base, Ohio, and other operating locations. He also directs oversight of more than 20 contractor support agencies. Michels has a bachelor's degree in management from Metropolitan State College, Denver, Co., a master's degree in logistics systems from Georgia College and State University, Milledgeville, Ga., and is a graduate of the Air War College, Maxwell AFB, Ala. He is also a graduate of the Defense Systems Management College's Program Management Course and holds Acquisition Professional Development Program Level 3 Certificates in both Program Management and Acquisition Logistics.

Bonnie Bollinger is a former GS-12, software engineer, and was assigned to the Avionics Directorate at Robins Air Force Base, Ga., prior to her retirement from federal service in 2001. The Software Engineering Institute, Carnegie Mellon College, Pittsburgh, Penn., has certified her as a lead assessor. Bollinger is currently a private consultant to SEI and government agencies.

Point of Contact: Stephen D. Acuff
226 Cochran Street, Robins AFB, GA 31098
Phone: (478) 926-6160 Fax: (478) 926-4911 E-mail: stephen.acuff@robins.af.mil

How Do I Make My Organization Comply With Yet Another New Model?

Sarah A. Sheard
Software Productivity Consortium

Comments often go something like this when organizations have to change their current process improvement model: "Help! We have to revise all our ISO 9000 processes to meet the Capability Maturity Model® (CMM®)!" Or, "I don't want to write systems engineering processes to EIA/IS 731 because I'm only going to have to rewrite them to CMM Integration℠ (CMMI℠)." If these sound familiar, it is because organizations are currently faced with the prospect of shifting their process improvement focus from a current model such as EIA/IS 731, ISO 9000, or the Software CMM to a new model such as the CMMI. To some, shifting seems a daunting feat; in many cases, it has taken years to adapt to the current model. Starting over seems to negate the value of the current investment. This article shows that capability models (and other process standards such as ISO 9000) generally ask the organization to do similar or identical things; adopting a new model should be approached as mapping and modifying rather than as starting over.

Many companies have begun documenting processes in response to ISO 9000 [1], one of the various capability maturity models, or perhaps to other standards [2, 3, 4]. These standards and models require documentation of, and disciplined adherence to, organizational processes. One concern is that these models are themselves changing. For example, an updated version of the ISO 9000 series of standards was released in 2000. Also, the Software Capability Maturity Model® (CMM®) (SW-CMM) [5] has been merged with the Electronic Industries Alliance/ Interim Standard (EIA/IS 731) [6] and with a draft version of the Integrated Product Development (IPD) CMM [7] to form the CMM Integration℠ (CMMI℠) [8].

When a new model comes along, it is normal for the process group, and even for management, to panic. With the abundance of standards and models currently available, and a tendency to want organizational processes that comply with everything, being a member of a process group is not always a calm experience.

Early ISO 9000 auditors used to tell companies to be sure that their Quality Manual follows the ISO 9001 structure exactly so there would be no difficulty getting registration. Auditors may not still make that kind of recommendation, but two of the problems that already have occurred if companies have taken such advice are:

- An ISO 9000 structure may make it difficult for the companies to also comply with the SW-CMM.
- The ISO 9000 standards have changed! Does this mean every company should now restructure its company's quality manual even though the manual is working well?

The question, therefore, becomes how to use a capability model in a manner that will work for years.

This article discusses the nature of capability models first, followed by descriptions of organizational processes and how they should be structured compared to the models. Integrated models and integrated processes are addressed as a special and optimal case. Mapping models and addressing the gaps is presented first in an abstract manner, and then suggestions are made that are specific for making the transition from systems engineering models to an integrated capability model.

"Increasingly, government agencies and their prime contractors are asking bidders to demonstrate maturity in their development efforts."

What Is a Capability Model?

Capability models define the characteristics of good processes and avoid prescribing how the processes must be enacted. The purpose of capability models is to establish a process improvement road-map upon which a route can be drawn from "where we are today" to "where we want to be." In order to determine "where we are today," an organization performs an appraisal, sometimes with the aid of an outsider with specific expertise in the model.

Capability models are not processes. They intentionally do not address a particular life cycle or sequence of activities. They also do not have the necessary characteristics of processes; namely, they do not include inputs, outputs, tasks, roles and responsibilities, and entry and exit criteria.

Rather, capability models are sometimes thought of as containing requirements for good processes. Capability models ask for processes to be written that have inputs, outputs, tasks, roles, entry and exit criteria, verification mechanisms, and measurements. These processes will be unique to the organization for which they are written.

Why Use a Capability Model?

There are several reasons why organizations use capability models:

- **To verify process contents.** Capability models encapsulate basic industry knowledge for an organization to use to help improve quality, customer satisfaction, productivity, and cycle time. Many companies examine the models to understand the basic practices: Is practice "X" necessary? If so, is it performed somewhere?
- **To demonstrate progress.** Another primary use of capability models is to demonstrate year-to-year improvement. Periodic ratings of the organization's processes against the models are one indicator of such long-term improvements.
- **To benchmark.** A model can be used to validate process improvement progress in comparison with competitors. Increasingly, government agencies and their prime contractors are asking bidders to demonstrate maturity in their development efforts. Companies that have received high ratings against a

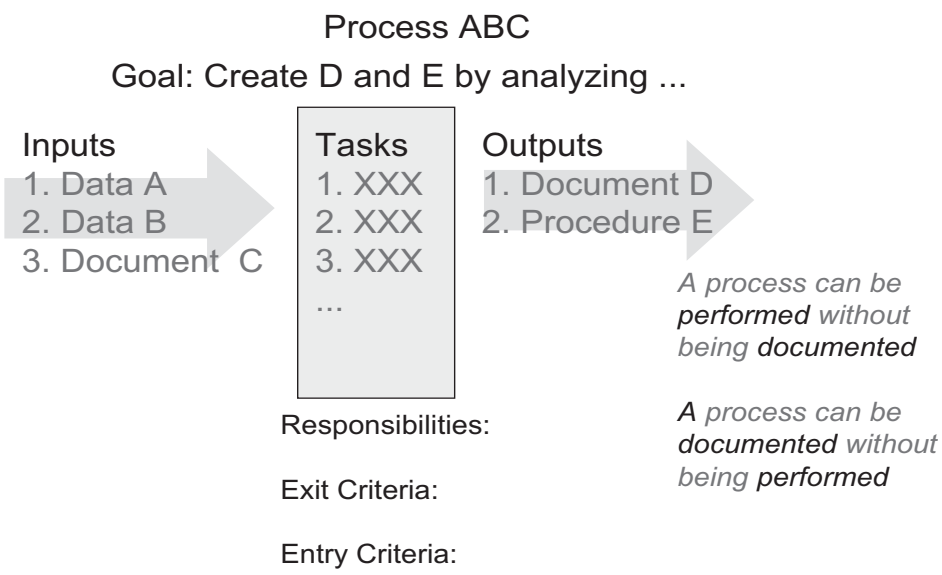


Figure 1: *Process Description Template*

well-known standard are considered to be more reliable, less risky contractors.

- **To structure new processes.** Organizations that have not yet captured their basic engineering practices in documented processes frequently will look at capability models as a list of what needs to be included.
- **Other uses.** Systems engineering capability models also can be used to help establish a definition of systems engineering to which the entire organization can subscribe. The structure of staged models defines a suggested road map for improvement.

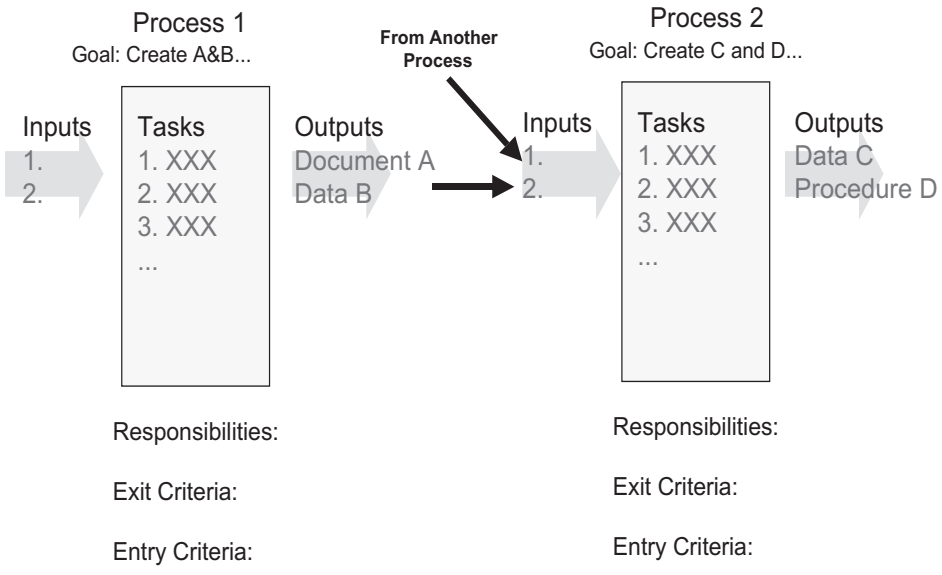
Structure of Capability Models

This section describes the structure of capability models from the point of view of systems engineering models.[2]

Continuous models contain between 17 and 24 process areas, also called focus areas or key process areas. These describe the activities that the processes must include in order to satisfy the model. The process areas are grouped into categories, but it is the process areas rather than the categories that are rated.

Models also describe how the process areas are performed, or capability. "Staged" models such as the SW-CMM include "common features," such as "commitment to perform, ability to perform, and verification," as a part of each process area. "Continuous" models such as EIA/IS 731 contain "generic practices" as a second axis, against which each process area is rated. Generic practices address the same aspects as the common features (e.g., resources, responsibility, verification, planning); in fact, some systems engineering models group generic practices into categories called "common features."

Figure 2: *Process Relationships*



What Is an Integrated Model?

The first capability model, the SW-CMM, addressed software development, or more precisely, the management of software development projects. Later models addressed systems engineering [9, 10], integrated product development [7], and other aspects ranging from human resources [11] to security [12]. Because these models targeted different functions, organizations often found themselves using multiple models separately in different areas of the organization.

An example is the Federal Aviation Administration (FAA). The Systems Engineering CMM (SE-CMM, a predecessor of EIA/IS 731), the SW-CMM, and the Software Acquisition-CMM (SA-CMM) [13] were in use in different departments. The FAA released the FAA Integrated Capability Maturity Model (FAA-iCMM)[3] in 1997 to unify its process improvement efforts [14]. The FAA-iCMM included every practice in every source model, as is shown by the detailed traceability tables included with each process area. The Software Engineering Institute (SEI) approved this model.

In late 2000 the SEI, serving as the custodian for a collaboration of industry and government groups, published the CMMI model. This model integrates most of the practices of EIA/IS 731 with the SW-CMM Version 2.0 (an unreleased expansion of the released Version 1.1) and much of the unreleased IPD-CMM. One version of the CMMI also includes some acquisition practices related to the SA-CMM.

What Good Is an Integrated Model?

Integrated capability models can do two things for the organization. First, they provide a common framework and terminology that encourages process engineering groups in the various disciplines to cooperate, both on the appraisals and also ideally on creating the organization's suite of processes. Such an integrated suite of processes can be significantly beneficial, as many of the process problems in organizations can be traced to poor interfaces between groups. Second, their associated appraisal methods can provide combined appraisals that cost less to perform than separate appraisals

What Is a Process?

A process is a sequence of steps to

achieve a given outcome.[4] Capability models require process descriptions to be documented. What is actually practiced must be available for reference and for instruction of newcomers.

Figure 1 shows a template for a process description. A process description should include the process goal, its inputs and outputs, a sequence of steps or tasks, and also preferably responsibilities or roles, and entry and exit criteria (when the process can begin, and when it is finished). Most process descriptions are textual, but for the visually oriented, a process diagram is helpful. Some organizations prefer to model processes in graphical or database-based tools.

How Are Processes Related?

Relationships among two or more processes should be described in a diagram that depicts the interactions among the processes (see Figure 2). For example, input needed by one process should be identified as the output of another process. Figure 3 shows the relationships of processes in a life-cycle engineering model surrounded by management and support.

Figure 4 shows an integrated systems and subsystems engineering process architecture. This architecture, based on the engineering "vee," shows system processes (Level n) and recursive sub-processes (Level n+1) such as those for subsystems, units, and components. Management processes perform overall planning and monitoring, and control baselines created in development processes as they are handed off from design to develop to integrate and test.

Several types of process architectures are available. The Software Productivity Consortium does not recommend a particular architecture. Organizations should structure processes in the way that best reflects that company's business.

What Are Integrated Processes?

As addressed above, disparate groups can initiate multiple process improvement efforts within an organization. Basing these efforts on different capability models and standards leads to disparate sets of processes such as software processes and systems engineering processes that neither integrate nor even have a defined interface. Unfortunately, integrated models do not provide guidance on how to integrate such disparate processes into a robust and optimized set of organizational processes.

There are several approaches to integrating these processes. Two starting points are expansion and integration. The following concepts should be addressed whatever the starting point.

Expansion

If only one group within the organization has documented its processes to date, developing an organization-wide integrated set mostly consists of expanding the scope of the documented processes and expanding participation in writing, reviewing, and using processes to additional groups. Note that the process architecture may require modification to be able to gracefully incorporate processes needed for the expanded scope.

Integration

If two or more groups have already begun documenting processes, there is more material with which to work but possibly more likelihood of misunderstanding. Do not approach the integration effort as trying to determine whose processes are better. Instead, analyze and discuss the following aspects of both sets of processes:

- **Boundaries.** What are the boundaries of the current processes? Do they encompass "everything the software department does" plus, say, "the engineering life cycle of a system, with emphasis on the contribution made by the program systems engineering group?" Are training, human resources, configuration management, program management, or engineering support environment maintenance covered in any way by either or both sets of processes? Draw out the overlaps and gaps.
- **Interfaces.** Next define the interfaces that should occur between the two sets of processes.[5] What should the software configuration management process provide to the systems engineering, or hardware configuration management process, and vice versa? Should they have the same data structures, or not? What data should pass from one group to the next? Is the software group expecting a complete and invariant group of requirements from the "systems engineers?"

Also note where the same processes seem to appear in both places: Requirements development and program planning are two typical examples. Should the software group and systems group plan the same way? Should they use the same data upon which to base task estimates?

Define the data in separate tables at first; for example, data that software is expecting to give and receive from other places vs. the systems group data. Clearly the systems group will be interacting with more than software; it's likely that the software group does the same. Where do the interfaces agree? Where do they disagree? What is missing from one data definition that is in

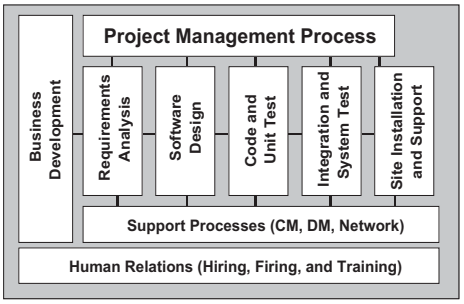
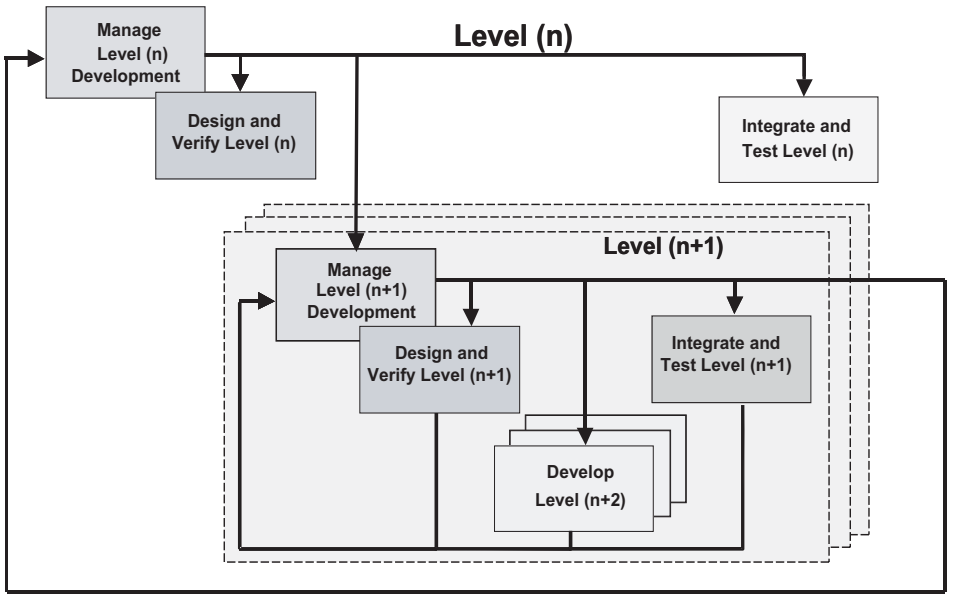


Figure 3: *Life-Cycle-Based Process Architecture*

Figure 4: *Integrated Systems and Software Process Architecture*



the other? It may be necessary to create joint tiger teams to resolve conflicts – better to resolve them early in the processes.

- **Participation.** Is the systems engineering group expecting software participation in determining what requirements should be allocated to software? Do (or should) systems engineers refrain from giving any requirements to software developers until the requirements are complete and stable? How does a systems change ripple through the systems and subsystem (e.g., software) processes? Work out these and similar issues in joint process meetings.

Integration Artifacts

The artifacts that will arise from the integration effort may include the following:

A) Processes that did not change or have minor adaptations:
- Software processes that will remain strictly software processes as they were before, e.g., code walk-throughs.
- Software processes that remain software processes but have to change their interfaces to match with systems, e.g., software requirements analysis.
- Systems processes that remain the way they were, e.g. factory acceptance tests.
- Systems processes that remain systems processes but have to change their interfaces to map with software and other processes, e.g., System Critical Design Review.
- Processes that don't exist yet in a documented form, but that's acceptable for now (other processes should acknowledge them and describe the assumed interface), e.g., program budgeting process.

B) Processes substantially reorganized to improve the effectiveness of the organization's set of standard processes:
- Systems processes that are no longer strictly systems processes; now merged with software or other subsystem area processes (or into other processes such as program management or configuration management processes), e.g., integration planning.
- Software processes no longer within the software group; now merged into systems or other processes, e.g., integration test processes or risk management processes.
- Processes that have to change to integrate better across the interface (perhaps delete one and use the other), e.g., requirements management.

C) New needed processes:
- Processes that need to be written now to address pressing organizational problems, e.g., a process for terminating a program, or for providing an engineering support environment

Adding Compliance to a New Model

It is time to look at how capability models and processes relate in order to determine how to add compliance to a new capability model. This requires understanding the whole picture and a fairly good understanding of three things:

1. The organization's processes.
2. The models with which the processes are currently compliant, and how the organization's processes relate to them. (Think of the models as the requirements, and the processes as the organization's solution. What is the traceability between the requirements and the solution?)
3. The model with which you wish to comply. (Think of this as another set of requirements with which your solution is already partly compliant.)

The goal is to understand where your existing processes do or do not comply with the new model. If you have documented traceability ("mappings") between the previous models and your processes, and between the new and previous models, you can infer mappings between your processes and the new model.

The bad news is that mapping is a lot of work for even one new model, and there are a lot of models out there.

The good news is that many models ask you for the same things such as "plan your work and work your plan," [15] so what you are doing is likely to transfer without much elaboration into compliance with a new model.

Consider the heritage of the new model. Especially if it is based on the models you have been using, the changes you will have to make in your organization's processes may be minimal.

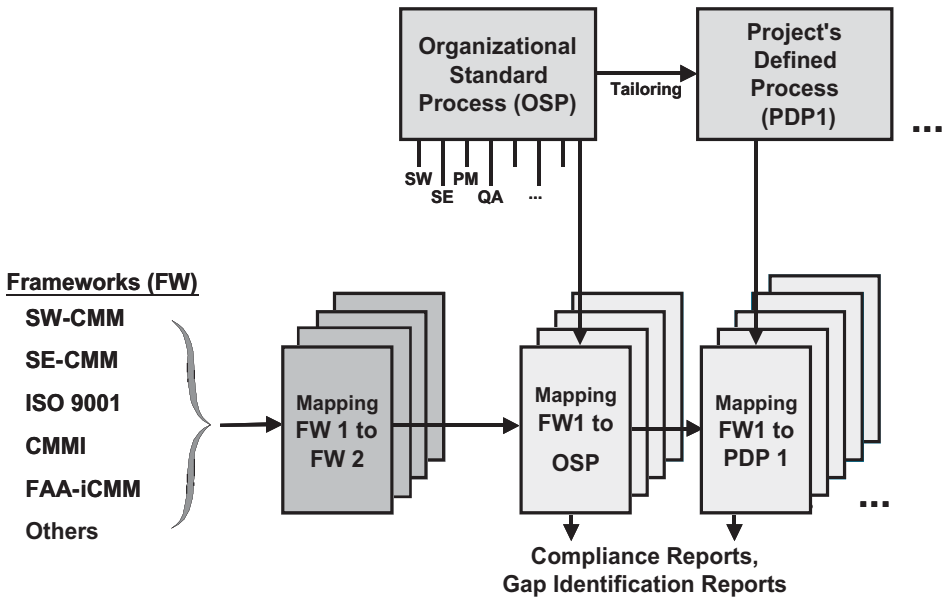
Mapping

When mapping to a single model, you are first setting up your processes to comply with a model; you need to know where in your processes you perform the practices that the model requires. A good mapping is two-way, meaning you could either look at an activity in a process and see what model practices this activity fulfills, or you could look at a practice in a model and see where these are included as activities in the organization's processes.

Once you have such a mapping, you should then continue mapping your processes to other models or standards of interest. Consortium members have access to a tool called Quagmap[6] that is pre-loaded with paragraph titles of popular models and shows how they map to each other. This allows input and mapping of an organizational set of processes to any of the preloaded models. Once a practice in one of your processes is mapped to a section of one model, the tool will give you an "inferred mapping" of the paragraphs in another model that also may map to this part of the process.

Figure 5 shows some of the many kinds of mappings, including mapping between the organizational standard processes and project processes. This kind

Figure 5: *Mapping Among Models and Processes*



of mapping, a record of project tailoring, is required for Level 3 in capability models.

When mapping processes to process areas, none of the capability models requires the processes of the organization to match the process areas of the model one for one. A process may combine the practices of two different process areas such as configuration management and data management, or planning and tracking. Several processes may be written to satisfy one process area (most organizations find a need to write more than one test process, perhaps unit test, integration test, and system test).

Gaps

A mapping between models should be able to highlight practices in the new model that were not well covered in the previous model. Concentrate on these gaps and address them on a case-by-case basis. Here are several ways to handle gaps:

- If your processes already include a practice because your business needs the practice, you may have covered the gap already.
- If your business truly does not need a new practice, you may be able to tailor out a requirement. For example, a system integrator may be able to omit practices related to implementing the design of a component.
- If the new model specifies additional practices that you do not currently perform but should, an activity might have to be added to (or changed in) one of your existing processes. An example might be certain measurement or risk management activities.
- If the old model did not include one aspect of your business that really does apply and should be improved, and the new one does, you might decide to write a new process to address this aspect (for example, appraisal of customer satisfaction).

You certainly do not, and should not, create a whole new set of processes to meet this new model.

It Sounds Like so Much Work

It is not as hard as it seems. The biggest difficulty in complying with a process improvement model is in making people understand what processes are, getting them written down, and getting management to enforce disciplined use of these processes. Specific activities that may be unique to a particular model are much less important than getting the organization to behave in a disciplined manner. Whether initial processes are configuration management processes, trade studies analysis processes, or even processes for running efficient meetings, it is easier to transfer discipline to additional processes than to perform processes in a disciplined manner in the first place.

Similarly, once organizationally standardized processes are understood, and programs know how their own processes differ from the standard processes, new programs begin to realize substantial savings in startup time and cost because the basic processes are already in place, and training is also easier. Adding new practices to the set of standard organizational processes, or even adding a few new processes, is easier than establishing an organization-wide process infrastructure in the first place.[7]

Moving From a Systems Engineering Model to an Integrated Model

The above suggestions address mappings in general. This section looks at specific changes an organization might need to make when the current model is a systems engineering model and the new model is an integrated capability model.

Maturity Rating

If you are used to continuous models and choosing process areas to emphasize, note that both the CMMI and the FAA-iCMM also include the concept of an official organizational rating, or maturity level, comparable to a rating in a staged model. An official Maturity Level 2 rating will require the achievement of the model's specified Level 2 process areas (and Maturity Level 3 requires both Level 2 and Level 3 process areas). Check to see if these are areas where you have put attention to date, and if you are seeking a rating, work hardest on any gaps.

Measurement

Measurement and the use of measurement data are called out much more explicitly in both integrated models than in the systems engineering models, and much earlier in CMMI (Level 2). This may be a surprise, but it is a good change. Work hard to comply with the measurement requirements and you will be rewarded with a much smoother process improvement effort afterward.

Process Improvement Infrastructure

Organizational process focus is much more clearly spelled out in the CMMI than in the systems engineering models. Systems engineering groups generally realize that they need a process infrastructure, but this process area, originally from the SW-CMM, helps define clearly what is needed.

Skills and Knowledge

The integrated models include process areas on training or organizational training. These may have a smaller scope than the Manage Competency focus area of EIA/IS 731, but may be more explicit in what must be done to comply. Review the practices against those of your processes that are mapped to manage competency.

More Good News

If you have reached mostly Level 2s or Level 3s in a systems engineering model, you will find you have most of the CMMI's requirements for comparable levels covered. This is much easier than starting over.

Some Questions to Ask

This section suggests some additional areas to investigate when adopting a new model.

Who does "systems engineering?" Do not assume the model considers "systems engineering" to be a particular organizational group. Most models do not specify who performs the systems engineering activities, just that they need to be done.

Do not make the mistake of assuming your "systems engineers" have to do everything, including configuration management, training, or program planning and tracking. While your systems engineers will need to be involved in these processes (managing the configuration of items they work on; requesting, taking, and possibly even planning training; and providing estimates and measurement data to program managers), many processes are run by groups other than those called "systems engineers." Furthermore, most subsystem areas, including software development, also have to perform systems engineering activities such as trade studies among potential architectures and validation against user need. Be sure they do not think they are exempt because the systems engineers do that.

How specific is the model? ISO 9000-2000 is fairly general, requiring certain aspects of processes to be well documented but not specifying much of their content (e.g., what is feasibility analysis). Some models such as EIA/IS 731 state that analyses must be done "as appropriate" while others, such as military standards and guidebooks, may explicitly specify a "feasibility analysis" deliverable and

include a required data item description.

How specific should your processes be? The temptation is to document processes by specifying all items in great detail so that any auditor can see the documentation is complete. This is not a good idea because (a) it will be more difficult to get engineers to follow 500 pages of documents than 50, (b) 500 pages is more expensive to write, (c) 500 pages is more expensive to maintain, and (d) some models, such as ISO 9000-2000, are strict about following that which is documented. This emphasizes that you should not build rigidity into a documented process if the business needs it to be flexible.[8]

Finally, most processes that have been in place for some time probably either meet most business needs, or else everyone knows they are broken. Look first at those places where your processes are broken. You may find that the model calls out performance of some practices that you do not do, and should. You can use the model as leverage.

Conclusions

Complying with a new capability model is much easier than starting fresh if the organization already complies with another model, particularly when the new model is an evolution of the old model. Especially if an organization's processes describe essential business practices and are mapped to the old model rather than strictly based on it, it is fairly straightforward to use other mappings to infer what parts of the processes may be lacking, according to the new model. Use your standard process improvement process to incorporate changes to meet the requirements in the new model. Other than your process group, the rest of the organization only has to understand the processes they are already using and the few changes; they do not have to understand the new model at all.◆

References

1. ISO 9000 International Standards for Quality Management. International Organization for Standardization, Switzerland, 1994 and 2000.
2. Sheard, Sarah A. "The Frameworks Quagmire." CROSSTALK, Sept. 1997 <www.stsc.hill.af.mil>.
3. Sheard, Sarah A., "The Frameworks Quagmire, A Brief Look." Proceedings of INCOSE. 1997b.
4. Sheard, Sarah A., and J. Lake, eds. "Systems Engineering Standards and Models Compared." Proceedings of INCOSE. 1998.
5. Paulk, Mark, B. Curtis, M. Chrissis, and C. Weber, eds. Capability Maturity Model for Software, Version 1.1. Software Engineering Institute, Carnegie Mellon University, 1993.
6. Systems Engineering Capability Model EIA/IS 731. Electronics Industries Affiliates, 1999.
7. An Integrated Product Development Capability Maturity Model, Version 0.9. Software Engineering Institute and Enterprise Process Improvement Collaboration, Carnegie Mellon University, 28 Oct. 1996.
8. Capability Maturity Model Integration, Software-Systems-IPPD-Acquisition, Version 1.02b. Software Engineering Institute, Carnegie Mellon University, 2000.
9. Bate, Roger, et al. Systems Engineering Capability Maturity Model, Version 1.1. Software Engineering Institute, Carnegie Mellon University, 1995.
10. Systems Engineering Capability Assessment Model, Version 1.5. International Council on Systems Engineering (INCOSE), 1995.
11. Curtis, Bill, William E. Hefley, and Sally Miller, eds. People Capability Maturity Model. Software Engineering Institute, CMU/SEI-95-MM-02, Sept. 1995.
12. Hefner, R., et al. Systems Security Engineering Capability Maturity Model. International Systems Security Engineering Association, 1999.
13. Software Acquisition Capability Maturity Model. Version 1.01. Software Engineering Institute, Carnegie Mellon University, 1997.
14. Ibrahim, Linda, et al. The Federal Aviation Administration Integrated Capability Maturity Mode, Version 1.0. Federal Aviation Administration, Nov. 1997.
15. Ferguson, JoAn, and Sarah A. Sheard, eds. "Leveraging the CMM for ISO/IEC 12207 Compliance." Nuts and Bolts, IEEE Software, Sept./Oct. 1998.

Notes

1. This paper originally appeared in the Proceedings of the Eleventh Annual International Symposium of the International Council on Systems Engineering, Melbourne, Australia, July 2001.
2. Other models place more emphasis on practices (parts of process areas) and goals (groupings of practices).
3. The FAA also released an appraisal method that includes an internal and an external "full appraisal" as well as several reduced-cost appraisal versions. This method is called the FAA-iCMM Appraisal Method or FAM.
4. This is a fairly narrow definition of process, chosen for understandability. Other broader definitions can be more abstract.
5. For process robustness, involve as many process users as possible in any restructuring of processes.
6. Quagmap is a trademark of the Software Productivity Consortium.
7. In fact, the most mature organizations have the easiest time of all. The essence of Level 5 is continuous improvement. At this level, inserting new or better processes is just the normal way of doing business.
8. A classic case of a process that needs to be flexible is how to develop a design that meets requirements. Some engineers think linearly and can do Step 1, Step 2, Step 3, but the best engineers often develop a picture in their head and cannot follow a stepwise procedure. Your processes should specify outputs and required tasks to produce the outputs, but should leave open how to do the tasks, particularly those that can be done by a single person, whenever possible.

About the Author

Sarah A. Sheard has 20 years experience in systems engineering, on systems from satellites to air traffic control. Currently at the Software Productivity Consortium, Sheard helps companies begin systems engineering process improvement efforts and integrate the software and systems engineering aspects of both development work and process improvement efforts. Sheard is currently the chair of International Council on Systems Engineering's (INCOSE's) Measurement Technical Committee and in the past has served INCOSE as chair of the Communications Committee and as programs chair of the Washington Metropolitan Area Chapter. Sheard received a bachelor's degree in chemistry from the University of Rochester and a master's degree in chemistry from the California Institute of Technology.

2214 Rock Hill Rd.
Herndon, VA 20170-4227
Phone: (703) 742-7106
E-mail: sheard@software.org

How Function Points Support the Capability Maturity Model Integration©

Carol Dekkers and Barbara Emmons
Quality Plus Technologies, Inc.

This article demonstrates the mutual effect of increased process maturity and an organization's maturity in their use of Function Point Analysis (FPA). As a company moves to a higher maturity level according to the Capability Maturity Model® Integration℠ (CMMI℠), its measurement maturity should also increase. The tie between the CMMI's process areas and FPA is not well understood, yet there is a direct connection that can be made between the model and FPA. The purpose of this article is to illustrate what those links are in terms understandable to non-experts in software measurement.

To better understand the relationship between the Capability Maturity Model® Integration℠ (CMMI℠) and Function Point Analysis (FPA), it is necessary to review the CMMI project. Readers already familiar with the CMMI can skip forward to the next section, "What are Function Points?"

The CMMI project began in 1998 as a collaborative effort of industry, government, and the Software Engineering Institute (SEI) to merge various SEI models [1]:

- Software Capability Maturity Model® (SW-CMM).
- Systems Engineering Capability Maturity Model (SE-CMM) (Systems Engineering published by Enterprise Process Improvement Collaboration: EPIC).
- The Systems Engineering Capability Appraisal Model (SECAM) (Published by International Council on Systems Engineering: INCOSE).
- Systems Engineering Capability Model (SECM) is the collaborative model of the SE-CMM and SECAM (EIA/IS-731).

In December 2000, a public review of CMMI V1.02 Models (consisting of CMMI-SE/SW/Integrated Product and Process Development [IPPD] V1.02 and CMMI-SE/SW V1.02) was announced and is currently under review. For more information on the SEI's CMMI model and the previous models, refer to <www.sei.cmu.edu/cmmi>.

Today's CMMI Model

The current CMMI is similar to its predecessor models in a number of ways. For example, the CMMI V1.02 Integration Systems/Software Engineering model has retained the five maturity levels of the SW-CMM and a set of corresponding process areas (formerly known as key process areas or KPAs) in the SW-CMM.

Level 1: Initial/Performed

Organizations appraised at Level 1 are characterized as simply "performing" software processes and do so in a manner that is often ad hoc or chaotic. The competence and heroics of the individuals doing the work drive how the activities are performed at this level. This first level in the CMMI model is the most common for development organizations embarking on process improvement initiatives. Processes defined at Level 1 are fundamental to software development.

> *"Function points are a technically independent measure that quantifies the size of functional user requirements of software."*

Level 2: Managed (Formerly called Repeatable in the SW-CMM) A Level 2 organization is more rigorous in both how it performs software processes and in the processes it performs.

At level 2, processes are managed: That is, they are planned, performed, monitored, and controlled for individual projects and groups, or they are standalone processes to achieve a given purpose. At level 2, managing the process achieves both the specific goals for the process area, as well as meets other goals such as cost, schedule, and quality.

Level 3: Defined

A Level 3 organization actually defines its processes and tailors them based on the organization's set of standard processes. Deviations beyond those allowed by the tailoring guidelines are documented, justified, reviewed, and approved.

Level 4: Quantitatively Managed

(Formerly called Managed in the SW-CMM) At Level 4, processes are controlled using statistical and other quantitative techniques. Quantitative objectives for product quality, service quality, and process performance are established and used as criteria in managing processes. Product quality, service quality, and process performance are understood in statistical terms and are managed throughout the life of processes.

Level 5: Optimizing

At Level 5, processes are continually improved based on an understanding of the common causes of variation inherent in processes. The CMMI SE/SW model identifies a consolidated set of process areas across the five levels. These will be discussed in further detail in the sections that follow.

What are Function Points?

Function points (FP) are a technically independent measure that quantifies the size of functional user requirements of software. The function point counting method used to calculate the number of function points is known as the International Function Point Users Group (IFPUG) function point method [2]. In its Counting Practices Manual release 4.1, IFPUG identifies the following objectives for function point counting:

- To measure the functionality the user requests and receives.
- To measure independently of implementation technology.
- To provide a normalization factor for software measurement.

Companies adopt function point size measurement in place of the traditional physical source lines of code because FP are independent of technology and implementation. For readers unfamiliar with FP, we commonly refer to them as the square feet of software because FP

quantify the size of the logical user[1] requirements. This is similar to quantifying a building's size by adding up the square feet from its floor plan.

IFPUG provides the following standard definitions in their FP Counting Practices Manual release 4.1 (1999):

> "Function point (FP). A measure, which represents the functional size of application software."
>
> "Function point analysis. A standard method for measuring software development and maintenance from the customer's point of view."

For further information about function points, refer to the IFPUG Web site <www.ifpug.org>, or see "Managing (the Size of) Your Projects" by Carol Dekkers in Feb. 1999 CROSSTALK.

What Is the Link?

The question remains "Where is the link between measurements, the CMMI's process maturity, and FP?" The SEI's CMMI SE/SW Model 1.02 presents a standard set of software processes to be satisfied before an organization can be appraised at a particular capability maturity level. Measurement, if used appropriately, can be a facilitator that can enable companies to move ahead and complete many of these process areas.

Measurement should encourage process repeatability: It is exactly the place where process maturity and FP coalesce to bring productive and positive bottom line results. The FPA counting process mirrors structured peer review to solidify "and" quantify the functional user requirements. As such, there are processes in the CMMI that could be supported by the process of documenting the logical functions counted with FPs, while other areas can be supported by the FP measure itself. In the latter case, the numbers and ratios based on the FP measure can be the most significant factor in achieving the process area. (See "Applying Function Point Analysis to Requirements Completeness" by Carol Dekkers and Mauricio Aguiar in Feb. 2001 CROSSTALK.)

While a picture can paint a thousand words, examples also provide illumination better than words. For example, in CMMI Level 2: Managed, the following Level 2 process areas (PAs) could be supported through FP based measurement:

1. Requirements Management PA:
   - Measurements can be made to assess the status of requirements – are they reviewed or accepted or rejected?
   - The amount of change activity can be captured – compare and quantify the FP size of change requests.
2. Project Planning PA:
   - Estimates for size of software work products (using FP) or changes must be derived according to documented procedure.

In another example, the CMMI Level 3 PAs include a formal software measurement program. In addition, repeatable processes can be supported with FP-based measurement. Traditionally, the capability maturity models did not explicitly mention measurement until Level 3: It was assumed to be a common theme underlying the process areas of every level. In March 2001, Dr. David Zubrow of the SEI revealed that the latest CMMI model now includes an explicit process area at Level 2 for measurement and analysis.

In 1998, Ken Dymond stated in his article *Using and Implementing the CMM*, "Many Level 1 companies fail to reach Level 2 … Lack of measurement to aid project planning, tracking and oversight, and requirements management. One of the most often missed measures is that of the size of various tasks/products to be performed/produced."

For the past three years, authors and leading experts have recognized the importance of software sizing in the CMM models. FP "size" addresses and quantifies software size objectively from a logical (user) point of view, independent of implementation, and is a valid sizing measure to use in conjunction with the CMMI.

FP Maturity Model

As an organization evolves to higher CMMI levels, its usage of FP-based measurements generally also evolves. This does not refer to the evolution of the function point methodology per se, but rather to how the organization leverages their FP counting process and makes use of their FP data and resultant measurements.

As briefly described in the previous section, FP can be used in almost all process areas of the CMMI, either as a part of a FP-based measurement, or the FPA process itself can be used to enhance the process area.

Goal-Question-Metric Measurement Approach

It is important to determine what FP-based measurements are appropriate for the current level (and capability) of your organization, as well as goals. As a case in-point, trying to implement Level 4 types of software measurement based on FP may be impossible, or at least frustrating, to do in an organization appraised as Level 1.

A good method for determining what measurements make sense for your organization at its current maturity is the Goal-Question-Metric (GQM)-approach. GQM provides that the goals of your organization and your measurement program be outlined first, before determining which questions need to be answered (to meet the goals), and which measures will answer the questions.

This is similar to determining requirements before building software – GQM sets up the requirements (goals) first, before setting the measurement details (the questions and metrics). Briefly, without getting into the depths of GQM, it is sufficient to say that goals must be SMART:

S - Strategic
M - Measurable
A - Achievable
R - Realistic
T - Targeted

(Note-These may be different than other uses of the SMART acronym.)

Ensuring that you have SMART goals before you decide on the questions and metrics to reach those goals will assist you to align your measurements with your own organizational capability (realistic, achievable).

FP-Based Measures for CMMI

This section illustrates what measurements and measurement processes can be used effectively at each level of the CMMI. If the FP process supports the process area, a "+" is indicated. Otherwise, it is the FP measure that supports the process area.

CMMI Level 1 – Initial/Performed

A maturity Level 1 is typically ad hoc and chaotic. For example, sample system measurements for Level 1 are "size of applications in FP." Sample software project measurements for Level 1 are "size of projects in FP."

CMMI Level 2 – Managed

These Level 2 PA support the inclusion of FP-based measurement:

- Requirements Management – Use FP to quantify the size of the functional requirements.
- Project Planning – Use FP to estimate effort and cost.
- Project Monitoring and Control – Use

FP to keep track of scope changes, percent complete, etc.
- Supplier Agreement Management – Use FP to define size of requirements (application/project being outsourced) and for monitoring supplier's progress, evaluating alternatives for outsourcing, and writing service level agreements.
- Measurement and Analysis – Collecting FP data, reporting data to managers/project managers, and understanding how FP can/cannot be used.
- Process and Product Quality Assurance – Use FP to track defects.
- Configuration Management – Use FP to track changes to requirements and to measure impact to project size.

These are sample system measurements for Level 2:
- Portfolio Size and Growth Trends – Percent increase/change of FPs from one date to another.
- Age of Applications – Percent of applications or percent of FPs older than five years, etc.
- Application Support Rates for Individual Applications – Number of FP per resource.
- FP by Language, Platform – Percent of FP Mainframe Cobol, etc.
- Application Churning – Number of change requests and FP size.

These are sample software project measurements for Level 2:
- Delivery Rates/Duration Delivery Rates – FP/hour or FP/month.
- Productivity – Hours per FP.
- Stability Ratio, Scope Creep/ Requirements Volatility – Percent FP added, changed, deleted.
- Project Cost per FP – Dollars per FP.
- Defect Ratio – Defects per FP.
- Testing Proficiency Ratios – Percent defects found pre-delivery/total defects (up to one month post-delivery) per FP.

CMMI Level 3 – Defined

These Level 3 PAs support the inclusion of FP-based measurement:
- Requirements Development – FP analysis process serves as a design walk-through of requirements.
- Technical Solution – Review FPA for design issues and for documenting interfaces and project documentation.
- Product Integration – Use FP to measure interface with other applications.
- Verification – Update FPs based on verification process.
- Validation – Track defect data and use FP as denominator.
- Organizational Process Focus+ – Include FPA as a standard organizational process.
- Organizational Process Definition+ – Include FP as part of the organizational measurement repository.
- Organizational Training – n/a.
- Integrated Project Management – n/a.
- Risk Management+ – Using FP to measure the size of the application will enable assessing risk changes when the size changes.
- Decision Analysis and Resolution+ – Using FP to assist in decision-making, i.e., impact of scope changes on schedule, extend schedule, increase resources, renegotiate scope, etc.

These are sample system measurements for Level 3:
- Application Support Rates for Organization – FPs per month.
- Support Activity Trends – Number of FPs per person over time.
- Application Maintenance Load per Person – FPs per person.
- Application Maintenance Cost per FP – Dollars per FP.
- Mean Time to Repair – Using FP to normalize based on application size.

These are sample project measurements for Level 3 (compare projects):
- Delivery Rates by Type of Project (development, enhancement, maintenance, language, platform, etc.) – Hours per FP.
- Duration Delivery Rates by Type of Project (elapsed duration) – FPs per month or hours per FP.
- Trending of Delivery Rates – FPs per month or hours per FP over time.
- Testing Proficiency Ratios for Organization – Percent defects found pre-delivery/total defects (up to 1 month post-delivery) per FP.
- Scope Creep/Requirements Volatility – Comparisons between projects, percent of change measured with FPs.
- Defect Density – Defects per FP.

CMMI Level 4 – Quantitatively Managed

These Level 4 PAs support the inclusion of FP-based measurement:
- Organizational Process Performance – Use FP as one of the measures for establishing performance baselines for processes.
- Quantitative Project Management – Use FP as the common denominator for selecting measures and statistical control of products and quality.

These are sample system measurements for Level 4:
- Enterprise Productivity Rates – Total FP/ total information systems work effort.
- Enterprise Quality Rates – Defects per FP compiled for the enterprise.
- Enterprise Cost Per FP – Dollars per FP compiled for the enterprise.
- Mean Time to Failure – Elapsed time/number of failures, normalized by FP.
- Mean Time to Repair – Elapsed time/number of failures, normalized by FP.

These are sample software project measurements for Level 4:
- Enterprise Project Delivery Rates – Hours per FP compiled for all projects.
- Enterprise Quality Rates – Defects per FP compiled for all projects.
- Enterprise Cost Per FP – Dollars per FP compiled for all projects.
- Statistical Process Control of Project Delivery Rates Trend Analysis of Projects – Using FP to normalize.

CMMI Level 5 – Optimized

These Level 5 PAs support the inclusion of FP-based measurement:
- Causal Analysis and Resolution – Analyzing defect data using FP as the common denominator enables comparisons between projects; allows evaluating impact of changes by using FP as the common denominator.
- Organizational Innovation and Deployment – Use FP as a measure for establishing process improvement objectives.

These are sample system measurements for Level 5 (all normalized by FP):
- Repair Cost Ratio.
- Defect Density – Defects/FP.
- Cumulative Defects.
- Defect Distribution by Severity.
- Defects by Cause.
- Mean Time Between Defects.
- Application Support Rate Trends After Process Improvements.

These are sample software project measurements for Level 5 (all normalized by FP):
- Defect Detection Ratio (by phase found). Also known as inspection effectiveness.
- Defect Removal Efficiency.
- Defect Distribution by Severity.
- Defects by Cause.
- Delivery Rate Trends After Process Improvements.
- Statistical Process Control of Defect Data.
- Cost of Defect Removal by Inspection Phase.
- SPC of Process Information.
- Post Release Defect Stability (by

month).
- Software Quality – Post-Release Defect Density.

Summary

In order to fully exploit and leverage the FPA benefits (the process of measuring "and" the measure) in a CMMI or process improvement environment, it is critical that you know what level your organization is. Understanding the capability maturity level of your organization will greatly assist you and your organization in establishing SMART goals, and applying appropriate FP-based measurements based on GQM priorities.◆

References

1. Software Engineering Institute, <www.sei.cmu.edu/sema/welcome.html>, <www.sei.cmu.edu/cmmi.html> for CMMI, and <www.sei.cmu.edu/activities/cmm/ acronyms.html> for acronyms.
2. International Function Point Users Group, <www.ifpug.org>.
3. Quality Plus Technologies, Inc., <www.qualityplustech.com>. Free articles on function points, software measurement, cultural issues, secrets of highly successful measurement programs, use cases, etc.
4. Software Technology Support Center. CROSSTALK, July 2000, <www.stsc.hill.af.mil/crosstalk>.

Note

1. The word "user" in the context of function point analysis refers to any person, thing, outside department, or other application that specifies the requirements for the software, or that has requirements to interact with the software. For a concise and non-technical discussion of a variety of terms with specific, non-traditional information technology meanings in function points, refer to Demystifying Function Points – Clarifying Common Terminology by Carol Dekkers, March 2001 available from Quality Plus Technologies, Inc. at <www.qualityplustech.com>.

About the Authors

Carol Dekkers, certified management consultant and certified function point specialist, is president of Quality Plus Technologies and is a recognized expert in the area of functional size measurement (Function Point Analysis). She represents the United States to the International Organization for Standardization as a project editor of their functional domain standard. The American Society for Quality (ASQ) honored Dekkers in 2000 as one of the 21 New Voices of Quality for the 21st century. Dekkers is the vice-chair of the Project Management Institute's Metrics Specific Interest Group, the past-president of the International Function Point Users Group, and a regional counselor for the ASQ Software Division. Dekkers is a frequent presenter at industry conferences, an author, and a weekly talk-radio host.

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

Barbara Emmons is a certified function point specialist with Quality Plus Technologies, Inc. She is a member of the International Function Point Users Group and currently serves on the Management Reporting Committee. She has more than 15 years of experience in information technology, over half of that in quality assurance or software measurement roles. In addition to consulting with companies to achieve software measurement and process improvement success, Emmons saw the successful integration of function points and the Capability Maturity Model® as the lead analyst at a prominent state government agency.

8430 Egret Lane
Seminole, FL 33776
Phone: (360) 402-0163
E-mail: beemmons@qualityplustech.com

U.S. Army Develops High Quality, Extremely Low Cost Digital Message Parser

Edgar Dalrymple
U.S. Army Aviation and Missile Command

This article describes a software project managed by the U.S. Army Aviation and Missile Command, Software Engineering Directorate (SED). The project was nominated to CROSSTALK *in its search for the top five software projects in the U.S. government. The project received high scores by the Software Technology Support Center review team for its description of the development of a highly automated process that the SED established to produce a digital message parser for the U.S. Army. The parser developed by the SED team, and their support contractor Science Applications International Corporation, has helped major weapon systems to demonstrate interoperability deemed critical to the future of the U.S. Army. The SED process has produced software of extremely high quality, and at the same time, extremely low cost. This article describes how these two characteristics, which may often seem mutually exclusive, were designed into the product by virtue of the development process that was established by the SED.*

Battlefield information has always been an important component of war fighting. However, the U.S. Army's vision of the future is a fighting force linked as an integrated collection of interoperable systems. Loosely referred to as digitization, the addition of software-based information systems to weapon systems is the means to achieve this goal. Graphical maps populated with icons representing enemy and friendly forces are coupled with obstacles such as mine fields and air defense artillery to provide what is called situational awareness (SA) data. Satellites, aircraft, ground vehicles, and individual troops collect this data. It is distributed via a wireless computer network know as the Tactical Internet (TI). The data is used for mission planning and execution, and greatly enhances the effectiveness of every individual system, and the Army as a whole.

One of the key digitization challenges centers on the definition and implementation of a standard message protocol. Historically, different families of weapon systems have preferred different protocols, selecting them based on the needs and limitations of their individual systems. Since there was no requirement, or even any mechanism for them to interoperate, there was no reason for them to share common protocols. To obtain the desired interoperability, a standard message protocol for use by all nodes on the TI was needed.

Toward this end, a digital message protocol known as the Joint Variable Message Format (JVMF) has been selected for use by the Army. In fact, all Department of Defense (DoD) systems and our allied forces will eventually implement the protocol. The joint designation means that the protocol will be multi-service. It is the variable format nature of the protocol that adds a large amount of complexity and difficulty for the software developers who are faced with its implementation.

> *"The specification of the messages is nested up to six levels deep, and each level can have potentially thousands of data elements that may or may not be present."*

A variable format was selected in part to minimize input/output (I/O) loading on the very limited transmission bandwidth that is available with current Army wireless technology. Standard Army radios have been improved to transmit voice and data, but have a nominal throughput of about 9600 bits per second. This capacity is a significant bottleneck in the transmission of graphics and messages that can reach into the hundreds of megabytes in size.

Message Specifications

The specification for the JVMF message protocol is the Technical Instruction Design Plan that is maintained by the Army's Communication and Electronics Command (CECOM). The specification is effectively maintained as a database that is known as the Variable Message Format (VMF) Integrated Database (VID). The VID defines the possible data fields and their associated parameters, structure, and the message cases and conditions. Cases and conditions are assertions about the consistency of the fields in the messages, and the parser must implement them in order to encode and decode a valid message.

CECOM produces a new database release when either new messages have been added, or existing ones have been changed. The specification of the messages is nested up to six levels deep, and each level can have potentially thousands of data elements that may or may not be present. The current version of the VID has 121 messages, with millions of fields possibly present. The information that could be contained in the full message set if all fields were populated would cause the storage size for the messages to be in the terabytes range.

By design, JVMF protocol eliminates the need to transmit empty, placeholder data packets that would be required in a fixed-format, character-based protocol. This means that only message fields that have valid user data – as signified by a bit known as the field presence indicator – get transmitted. This reduces the transmission load on the hardware. However, the complexity of the variable format places very stringent requirements on the message parsers that encode the user data into properly formatted binary data, and then subsequently decode the binary data back into user data on the receiving end. If one bit is set incorrectly, the entire message is unreadable by the recipient. Also, as the specification of the message types and field definitions is expected to continually evolve, some early studies have deemed the protocol unworkable based on the assertion that the message parsers would be too com-

plex to build, test, and maintain.

First to Face Digitization

The parser project was initially established in support of the Bradley A3 Fighting Vehicle, managed by the U.S. Army Tank and Automotive Command (TACOM). The A3 was one of the first major Army systems faced with digitization. While working with TACOM and its prime contractor, United Defense Limited Partnership, the Software Engineering Directorate (SED) anticipated and solved the problems associated with the development of the parsers by using sophisticated software engineering methodologies commonly used in compiler development.

The methodology for the development of the JVMF message parser was designed to solve the following problems:

- Produce software of exceptionally high quality.
- Maintain very short cycle times for product release.

The means by which these characteristics were achieved included the following:

- Maximum use of automation.
- Use of a Software Engineering Institute Capability Maturity Model® Level 3, and later Level 4, process.

The basic approach to the solution involved the following steps:

- Describe each of the messages in the VID in a formal Backus-Naur Form (BNF) grammar. This allowed for an unambiguous specification of the message structure.
- Develop automated software tools to read the VID and directly produce the BNF grammar.
- Develop automated software tools to read the BNF grammar and produce the Ada source code for the message encoder and decoder, i.e., the parser.
- Develop a test case generator to read the database and produce the suite of test cases to fully test all message fields for valid data, data ranges, and cases and conditions checks.

The SED defined a two-level BNF grammar. This allowed the entire (i.e. semantic and syntactic) behavior of the parser to be formally described. The fact that a two-level grammar was defined was essential to producing a parser that was capable of performing the cases and conditions checking. Not doing this would have made implementing the checking a manual activity. Development and maintenance of the code verifying cases and conditions by manual means would have made achieving the quality and cycle time goals unobtainable. Failing to implement the cases and conditions checking at all would have produced an unacceptably inferior product.

To support the quality and automation goals, the team defined and developed software tools wherever possible to automate every feasible step of the process. This ensured a highly repeatable process was put in place. Once the tools were mature and debugged, it proved to be a process capable of producing extremely high levels of quality.

A brief summary of some of the tools and their functions follows. These tools are primarily written in the Ada programming language.

- Data Extraction Tool: Extracts the message specification data from the database supplied by CECOM to create text files used by other tools.
- Grammar Generator Tool: Reads the text files produced by the data extraction tool and produces the BNF grammar to represent the messages.
- Parser Generator: Reads the grammar generated by the grammar generator and produces the Ada source code files that represent the message parser for the messages.
- Test Data Generator: Generates more than 6,000 test cases to test the parser.

New tools are added as the customers' needs demand. For instance, a Symbol Table interface to the parser was created for the Program Executive Office of the Aviation Electronics Command. The symbol table parser was implemented in the Improved Data Modem (IDM). This was done to isolate software changes to the IDM, thus avoiding frequent changes to interfacing components in the avionics systems of helicopters. The flight certification concerns and associated costs due to changes to avionics software make this a very important benefit to IDM customers.

Product Performance

Currently the IDM is used in three helicopters. The ability to use the same parser software in three systems has saved an estimated $1.2 to $1.5 million dollars versus the cost of redeveloping similar code for each system. If used on an additional fourth aircraft as planned, an additional $2 million dollars in savings is projected. There is also an unquantified cost savings due to the quality of the product. Attempting to integrate unreliable, low quality software into an embedded, real-time system can have disastrous consequences on cost, schedule, and product quality. The users of this product have come to trust the SED development process to support their cost and schedule with some of the highest quality code in the Army.

The following metrics reflect the performance that has been achieved by the project. The Ada source code for the parser as counted by the terminal semicolon method consists of approximately 204 thousand lines of code. The duration required to fully generate, test, document, and deliver this code is three months from the receipt of a valid database. Engineering releases can be done in two to three days. The effort required is four person months, based on 152 person hours/person month. The post release defect density of the last three releases, as measured for six months after the commencement of operational use is 0.1 defects per 1,000 single line of code (SLOC). The current cost of the delivered product, ignoring the costs of the initial development of the tool-set, is approximately $0.04 per SLOC.

The product is delivered with a full set of documentation, including requirements documents, test plan and procedures, application program interface specification, interface control document, version description document, Ada source code, and C language bindings. Users can compile and link only the encoder and decoder files that are needed by their application, thus avoiding the overhead of a solution that consists of a standard executable. The C language bindings enable the product to be used on virtually every hardware platform used by the Army.

As a final measure of product quality, the product is tested using the CECOM supplied VMF Test Tool (VTT). The VTT is the official test tool that provides certification of correct implementation of the message protocol. The IDM was independently tested by the CECOM labs and found to be in full compliance with the message standard.

Conclusion

This project provides a key technology to support one of the most significant organizational goals in the Army, interoperability via digitization. The increased SA data provides for a major increase in the effectiveness of Army systems. The highly automated develop-

ment process allows for extremely short cycle times in the generation and release of the product. This allows weapon systems to quickly update their ability to interoperate and provides the war fighting advantages of increased SA.

This would not be possible without the process the team has put in place to support this product. The methodology employed on this project was sophisticated and targeted to the risk associated with the application. It is a textbook example of how to craft a solution to a problem based on risk. The product speaks for itself. Hardly a week goes by that some other project fails to call and request a copy of the parser.

The fact that the technology is government owned allows for it to be distributed free of charge to DoD programs. The relatively small cost of maintaining this product is borne by the SED to the benefit of the entire Army. Custom enhancements, such as the symbol table interface, are funded by customers, notably IDM product office.

This project is an outstanding example of how the Army Life Cycle Software Engineering Centers can be used to benefit a wide range of Army systems in a cost effective and high quality manner.

Key Players

The most important part of any project is its members. The author would like to specifically mention the following team members who have been key to this program:

- James Magnusson, Science Applications International Corporation (SAIC) assembled the technical team.
- Doris Chan, SAIC, has served as the parser technical lead for six years.
- John Shannon, SAIC, developed and defined the methodology to employ the formal BNF grammar.

All of these people have more than 20 years experience in the design, development, and management of embedded software systems for military applications:

- Larry Stanbery, SAIC, enhanced the automation of the testing process and has more than 10 years of software experience.
- Charles Hyder, SAIC, provided software quality assurance support.
- Debra Henry, EER Systems, provided configuration management support.◆

About the Author

Edgar Dalrymple is a lead project engineer and division manager at the U.S. Army's Aviation and Missile Command, Software Engineering Directorate. For 15 years, he has supported major Army systems in the areas of software quality assurance, independent verification and validation, and software project management. In addition to the Joint Variable Message Format message parser, he currently supports the Bradley A3 modernization. He received a bachelor's degree in nuclear engineering from North Carolina State University in 1979.

U.S. Army Avionics and Missile Command
AMSAM-RD-BA-SQ
Redstone Arsenal, AL 35898
Phone: (256) 842-0187
E-mail: edgar.dalrymple@sed.redstone.army.mil

WEB SITES

Note: *Newcomers to software process improvement or the Capability Maturity Model® Integration℠ will benefit from these Web sites that offer information and training on all aspects of software process improvement.*

Software Technology Support Center

www.stsc.hill.af.mil

The Software Technology Support Center (STSC) 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. The STSC is offering a free workshop to U.S. government employees on Capability Maturity Model Integration Version 1.1 on May 21-23, 2002.

Software Technology Conference

www.stc-online.org

The Software Technology Conference (STC) is the Department of Defense's premier software conference. One tutorial and one track will be specifically aimed at addressing the Capability Maturity Model Integration Version 1.1. The CMMI tutorial can be found on Monday, April 29, 2002 Track 4 at 1:00 p.m.; the CMMI track can be found on Wednesday, May 1, 2002 Track 4 at 1:00 p.m.

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 to improve their software engineering management practices. The site features complete information on models it 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.

Software Productivity Consortium

www.software.org

The Software Productivity Consortium (SPC) is a nonprofit partnership of industry, government, and academia. The SPC 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. Membership is open to all U.S. or Canadian-based companies, government agencies, and academic organizations.

THE FOURTEENTH ANNUAL

Software Technology Conference

Forging the Future of Defense Through Technology

29 APRIL - 2 MAY 2002 • SALT LAKE CITY, UT

There is great wisdom in relying on the technology that produces the strength of a "drop-forged" tool. Just as hammering metal makes it stronger, our defense systems will be made stronger through the application of technology. Technology can be seen as the hammer we will use to shape the future of defense. The Software Technology Conference (STC) will focus on how we can best use technology to strengthen the future. This year's conference is shaping up to be the best yet. Join us in sharing your knowledge, opinions, research, and lessons learned and help us in "Forging the Future of Defense Through Technology."

About STC

The best way to describe STC 2002 is that it is jam-packed! This year's conference will include more than 180 events to choose from, including general sessions, luncheons, plenary sessions, and presentation tracks. If you work with software, STC provides outstanding training and networking opportunities. Some organizations report that they must send a small army to absorb all the information that is important to their organizations.

In its fourteenth year, STC is the premier software technology conference in the Department of Defense and is co-sponsored by the United States Army, United States Marine Corps, United States Navy, United States Air Force, the Defense Information Systems Agency (DISA), and Utah State University Extension. We anticipate over 3,000 participants this year from the military services, government agencies, defense contractors, industry, and academia.

General Sessions, Panel Discussion, and Plenary Speakers

The STC Opening General Session will be held Monday morning of conference week and will feature two keynote speakers: **Congressman James V. Hansen**, Republican, 1st District of Utah, and **Mr. Lloyd K. Mosemann, II**, Senior Vice President for Corporate Development, SAIC. The co-sponsors will host a panel discussion Tuesday morning, moderated by **Ms. Dawn C. Meyerriecks**, Chief Technology Officer for the Defense Information Systems Agency.

Wednesday's plenary session speaker is **Mr. Kevin Fitzgerald**, Senior Vice President and General Manager, Oracle Corporation. Thursday, **Mr. Grady Booch**, Chief Scientist, Rational Software Corporation, will address the conference. STC will be capped off Thursday afternoon with **Mr. Tom Talleur**, Managing Director of Forensic and Litigation Services for KPMG LLP, as the Closing General Session keynote speaker.

Special Sessions

Sponsored track presentations will be offered throughout the week by the following organizations: CinC Interoperability Program Office (**CIPO**), Computer Resources Support Improvement Program (**CRSIP**), Defense Information Systems Agency (**DISA**), Earned Value (**EV**), Institute of Electrical and Electronics Engineers (**IEEE**), International Council on Systems Engineering (**INCOSE**), Joint Strike Fighter (**JSF**), Office of the Secretary of Defense (**OSD**), Software Engineering Institute (**SEI**), and Software Technology Support Center (**STSC**).

STC is Endorsed by:

Lt Gen John L. Woodward, DCS/Communications and Information, U.S. Air Force

Lt Gen Harry D. Raduege, **Jr.**, Director, Defense Information Systems Agency

LTG Peter M. Cuviello, Director of Information Systems for Command, Control, Communications, and Computers, U.S. Army

RADM Kenneth D. Slaght, Commander, Space and Naval Warfare Systems Command, U.S. Navy

Dr. Donald C. Daniel, Deputy Assistant Secretary for Science, Technology, and Engineering, U.S. Air Force

Mrs. Debra M. Filippi, Deputy Director of Command, Control, Communications, and Computers, U.S. Marine Corps

On the Agenda

- Alternative Methods to Software Development
- Business: Evolution
- Capability Maturity Models
- Common and Open Systems Architectures
- Data Management
- E-ducation
- Earned Value
- Enterprise Software
- Higher Levels of Process Maturity
- High Integrity Software
- Information Assurance
- Interoperability
- Lessons Learned
- Process Improvement
- Project Management
- Quality/Quality Assurance
- Requirements
- Software Architectures
- Software Engineering Best Practices
- Software Intensive Weapon Systems
- Test
- Tools
- XML
- Wireless

STC 2002 Exhibiting Organizations (As of 11/16/01)

Ada Core Technologies, Inc.
AFIT Software Professional Development Program
Auburn University
Battelle
BEA Systems, Inc.
Boeing Co.
DDC-I, Inc.
Defense Contract Management Agency
DISA
Franklin Covey Government Products Group
Galorath, Inc.
GTSI Corp.
Hewlett-Packard Co.
Hill AFB
IBM/Lotus/Tivoli
Integrated System Diagnostics, Inc.
Jacada, Inc.
JAWS S3 2002
JOVIAL Program Office
Lockheed Martin
MCTSSA
MERANT
Microsoft Corp.
NASA GSFC SEWP
Northrop Grumman Information Technology
Novell
Objective Interface Systems, Inc.
OO-ALC/TIS
PHI Enterprises, Inc.
pragma Systems Corp.
Praxis Critical Systems, Ltd.
Quality Plus Technologies
Quantitative Software Management, Inc.
Quest Software
Rational Software
Real-Time Innovations, Inc.
RS Information Systems, Inc.
SAIC
SAS Institute, Inc.
Scitor Corp.
SEEBEYOND
SilverStream Software, Inc.
Software Configuration Solutions, Inc.
Software Dimensions & International Institute for Software Testing
Software Engineering Institute
Software Productivity Consortium
SPAWAR
STSC
Sun Microsystems, Inc.
SYNERGEN, Inc.
Top Graph'X
United Defense
United States Air Force
USACECOM SEC
Utah State University Extension
ViON Corp.
Vitech Corp.
Wind River

Special Events

STC 2002 features many networking opportunities such as the Opening Welcome Reception on Monday, and the Drag 'n Drop Social and Hotel California "A Salute to the Eagles" musical entertainment on Wednesday. Space for these events is limited.

Traveling with a companion? We are excited to offer two optional tours at this year's conference: an afternoon walking tour of the newly completed Gateway Center and an evening dinner tour at Gardner Village, both on Tuesday. You may purchase tickets to both of these optional tours as well as a companion package which includes admittance to selected conference events.

Registration

Completed registration form and payment must be received by 25 March 2002 to take advantage of the early registration fees. Credit cards will not be charged until 1 April 2002. The conference fee structure for STC 2002 is as follows:

Discounted registration fee (paid by 25 March 2002):

Active Duty Military/Government*	$595
Business/Industry/Other	$725

Regular registration fee (paid after 25 March 2002):

Active Duty Military/Government*	$665
Business/Industry/Other	$795

* *Military rank (active duty) or government GS rating or equivalent is required to qualify for these rates.*

The Housing Bureau of the Salt Lake Convention and Visitors Bureau (SLCVB), using the online Passkey system, handles housing reservations. Housing has been available since May 2001; therefore, some government rate guestrooms at specific hotels may not be available. To access the Passkey system, log on to the STC Web site at www.stc-online.org and select the Housing Reservation button. If you prefer to make your reservation using a traditional method, a PDF version of the housing form is available online.

Delta Airlines Special Discounted Airfare

Delta Airlines is our official host airline for all STC 2002 attendees. Take advantage of the five percent discount off Delta's published round-trip fares within the continental U.S. A ten percent discount is offered on Delta's domestic system for travel to STC 2002 based on the published unrestricted round-trip coach (Y06) rates. Book your flight by calling Delta Meeting Network® Reservations at 1-800-241-6760, Monday-Sunday 8:00 a.m. – 11:00 p.m. Eastern Time, or have your travel agent call for you. You must refer to File Number 181932A when making your reservations.

Trade Show

STC 2002 will again feature its accompanying trade show, providing 180+ exhibitors the opportunity to showcase the latest in software and systems technology, products, and services. This year's schedule has been adjusted to allow participants more time to interact with the vendors without conflicting with conference presentations.

Exhibit space is sold in increments of 10' x 10' at a rate of $1575 per 10' x 10' space if application is received on or before 15 February 2002. Should space still be available after this date, booth space shall be processed at the rental rate of $1775 per 10' x 10' space. Special fees and restrictions may apply to certain types of booth space. Complete trade show rules, regulations, and updated hall layout are available on the STC Web site.

All badged exhibit personnel wishing to attend the entire conference are eligible for a discounted conference registration fee. Please utilize the conference registration form that was mailed to the exhibit manager in early January to register for the conference.

www.stc-online.org

Source Code: CT6A

General Information
stcinfo@ext.usu.edu
435.797.0423

Technical Content Inquiries
stc@hill.af.mil
801.777.7411

Trade Show Inquiries
stcexhibits@ext.usu.edu
435.797.0047

Media Relations
stcmedia@ext.usu.edu
435.797.1349

LETTERS TO THE EDITOR

Dear CROSSTALK Editors:

Having read the November 2001 issue of CROSSTALK, I am somewhat at a loss concerning the section "Dynamic vs. Static Invocation" in the article *Factors to Consider When Selecting CORBA Implementations* by Dr. Thomas J. Croak. The author must have misunderstood the meaning of static invocation or he is using the term in a (for me) unknown way.

The author states that "static invocation can be used if the language, compiler, and operating system (and hardware) are known to be the same on both client and server." This statement is of course true, but certainly static invocation can be used regardless of differences in client and server operating systems, compilers, and languages.

Also in the section "Questions for the CORBA ORB Salesman," it is stated that "..., and you run the risk of future software failures given an operating system upgrade on portions of the architecture [when using static invocation]." I fail to see how this can be true, given that the CORBA architecture is specifically designed to be platform- and language-independent. Clients do not have to know, and indeed cannot know, implementation details of a server on the basis of the interface definition, and so will not be affected by server-side implementation changes.

According to my textbooks and to the best of my knowledge:

- Static invocation is used when you compile a client stub from Interface Definition Language and use that to contact the server skeleton. It is also known as early or compile-time binding. In other words, the interface (but not the implementation) is known beforehand.
- Dynamic invocation is used when the client does not know the interface in advance but queries the ORB for an interface (or method) definition. The client then builds a request based on the obtained definition and invokes it. This is known as late or run-time binding.

The difference between static invocation and dynamic invocation is very analogous to the difference between directly calling into vtable method pointers or using IDispatch.Invoke (after having called IDispatch.GetIdsOfNames and IDispatch.GetTypeInfo and so on ...) in COM.

Either way, the client will always be insulated from the server implementation details.

According to the glossary of the CORBA 2.5 specification (September 2001):

- Static invocation: Constructing a request at compile time. Calling an operation via a stub procedure.
- Dynamic invocation: Constructing and issuing a request whose signature is possibly not known until run-time.

Best Regards,

— Jan Holst Jensen

Dear CROSSTALK Editors:

The term "invocation" is overloaded to mean both the determination of the interface representation and the actual method call. The reader is using the first meaning of the term while I was using the second, and specifically stated so. The interface representation can be determined at compile time (referred to in the CORBA specification as Static Invocation Interface) or at run-time (referred to as Dynamic Invocation Interface in the CORBA specification). The method call on an object can be "invoked" dynamically or statically. In this sense of the term, the CORBA specification as well as other texts are not particularly clear. The interpretation I was using is that a dynamically invoked method call implies that the data will be marshaled, and a statically invoked method call implies that the data will not be marshaled. A danger of such ambiguity is that it leaves certain points open to the interpretation of the object request broker (ORB) provider, who may not implement the ORB consistently with the specification's intention.

The standpoint of my article dealt with safety-critical embedded and command-and-control systems typical of those found in Department of Defense combat systems where a function must be deterministic. Dynamic invocation, in either sense of the term would never be considered to be deterministic. Static invocation can be deterministic subject to the actual construction of the operating system, the compilers, and the ORB itself. However, the static invocation can become non-deterministic should any of these change. When I said, "Static invocation can be used if the language, compiler and operating system (and hardware) are known to be the same on both client and server," I should have said, "Static invocation can be used *safely* if …"

Speaking from experience, I can attest that the CORBA architectural philosophy of platform and language independence has not been faithfully extended to all ORB implementations, for all platforms and all languages. If you have an environment of mixed operating systems and languages, it is often difficult to find a single ORB vendor for the whole environment. If you must use multiple vendors' products, you will rarely have interoperability problems with the standard data types such as integer, real, or string. Try passing a scalar array or a covariance matrix and there will often be interoperability problems. When I said, "You run the risk of future software failure given an operating system upgrade on portions of the architecture," I was pointing out that an ORB supplier might make design decisions based on specific operating system functionality. If the underlying functionality of the operating system changes, the ORB's behavior may also change. When an ORB is ported to a different operating system or operating system version, testing does not always uncover all the behavior nuances, which may ultimately affect performance.

I feel strongly that any architect or designer should carefully consider the implications of any design decision on the expected behavior of the system as a whole, as well as the potential for unexpected changes in behavior given typical system evolution in compilers, operating systems, and commercial off-the-shelf products.

— Tom J. Croak

The Captain and the Kid

When was the last good fight in the software industry? Bill Gates versus the government – Boring. Napster was over before it started. Capability Maturity Model® (CMM®) versus, …well no CMM? We haven't had a good intellectual mêlée since Ada Wars.

Still, the air is thick with anticipation – an old fashion donnybrook is on its way.

With the help of Elton John and the words of Bernie Taupin, slightly customized, let's explore the skirmish *(It helps if you sing the Italic sections).*

It's getting late have you seen my mates
Ma tell me when the boys get here
It's seven o'clock and I want to rock
Want to get a belly full of beer [1]

In government's corner the Process Champion – weighing in on everything – Capability Maturity Model® Integration℠ (CMMI℠).

Captain Fantastic raised and regimented, hardly a hero
Just someone his mother might know
Very clearly a case for process and models
"Two teas both with sugar please [2]

Backed by the top-down management approach of the Software Engineering Institute.

But the biggest kick I ever got
Was doing a thing called the CMM Rock
While the other kids were Coding round the clock
We were hopping and bopping to the CMM Rock

Well CMM Rocking is something shocking
When your level just can't keep still
I never knew me a better time and I guess I never will
Oh Lawdy mama those Friday nights
When SEPG wore her process tight
And the CMM Rocking was out of sight [3]

In the People's corner – the Technology Champion weighing very little – Agile Software Development (ASD).

You know you can't hold me forever
I didn't sign up with you
I'm not a process for your friend's appraisal
This boy's too young to be singing the blues [4]

Backed by a grass-roots manifesto of simplicity, communication, and courage.

So goodbye maturity road
Where the dogs of piety howl
You can't plant me in your model
I'm going back to my code
Back to the howling old owl in the woods
Writing the thorny back code
Oh I've finally decided my future lies
Beyond the maturity road [4]

You don't believe it? Check out "CTO Diaries" in the January issue of *Software Development Magazine*, Jack Ganssle's article in the December issue of *Embedded Systems Programming*, or the November/December issue of *IEEE Software*.

Don't give us none of your aggravation
We had it with your discipline
Saturday night's alright for fighting
Get a little action in [1]

In the November/December issue of *IEEE Software*, Mark Paulk claims that eXtreme Programming can work well with the CMM.

Too late to save myself from falling
I took a chance and changed your way of life
But you misread my meaning when I met you
Closed the door and left me blinded by the light

Don't let the sun go down on me
Although I search myself, it's always someone else I see
I'd just allow a fragment of your life to wander free
But losing every process is like the sun going down on me [5]

Maybe Mark has something? After all, CMM is all about managing software processes. eXtreme Programming and its agile cousins are focused on developing software. Why couldn't you use agile software development and still be a CMMI Level 5? Actually, you could tan leather and be a CMMI Level 5.

We've thrown in the towel too many times
Out for the count and when we're down
Captain Fantastic and the Brown Dirt Cowboy
From the end of the world to your town [2]

Are you kidding? We can't get along because it's not about process or technology. It's about money. Few have the financial backing to concomitantly invest in higher levels of maturity and agile software development. Even if you could, who has the intestinal fortitude to balance the complexity of CMMI with the simplicity if ASD? What takes priority, predictability or agility?

The battle is coming. As with Ada, there will be no clear winner or loser but we will learn more about our organizations, profession, and ourselves, and we will be better for it.

The Captain and the Kid stepping in the ring
From here on sonny, it's a long and lonely climb [2]

— Gary Petersen, Shim Enterprise, Inc.

1. Saturday Night's Alright, Elton John & Bernie Taupin, © 1973 Dick James Music Limited.
2. Captain Fantastic and the Brown Dirt Cowboy, Elton John & Bernie Taupin, © 1975 Big Pig Music Limited.
3. Crocodile Rock, Elton John & Bernie Taupin, © 1972 Dick James Music Limited.
4. Goodbye Yellow Brick Road, Elton John & Bernie Taupin, © 1973 Dick James Music Limited.
5. Don't Let The Sun Go Down On Me, Elton John & Bernie Taupin, © 1974 Big Pig Music Limited.

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

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