CrossTalk

The Journal of Defense Software Engineering

Learning Best Practices

Sponsored by the
Embedded Computer
Resources Support
Improvement Program
(ESIP)

October 1999
Volume 12 Number 10

STSC

Published by the
Software Technology Support Center

Practice Makes Perfect

Tracy Stauder
Software Technology Support Center

When I hear the term practice, visions of my childhood emerge. Practice, practice, practice are the three words I often heard when it came to my extracurricular activities. I am sure most of you can relate. For example, did you take swimming lessons, tennis lessons, or attend basketball or baseball camp? Did your teacher or coach urge you to practice at least one hour a day? And do you remember your parents saying, "You can't expect to improve (or make the team) without practicing." Now that I am a parent, I also hear myself saying such things to my daughter as she takes tumbling and dance lessons.

Practice is a good thing. It makes us better performers. But what do you remember most about having to practice? It takes time, commitment, and energy. And didn't it seem that you always had something better to do, like watching television or playing video games?

However, I remember being envious of those that were the best. I wished I had the stamina to practice more and to be a better performer. I could only dream of what it would be like to make it to the Olympics. As we all know, the best of the best make it to that level. Whether it is swimming, diving, track, baseball, gymnastics, ice skating, or skiing, these athletes endure endless hours of training and practicing. Talent only takes you so far; it is the practice that makes you perfect.

I see similarities in the defense software community. Most software managers working with tight schedules and budgets cringe at the time it can take for them and their teams to learn and implement best practices such as configuration management, risk management, metrics-based scheduling, inspections, or defect tracking. They question whether all this hard work and practice will pay off.

In a sports scenario, you can think of software managers as the coaches, software engineering process groups as the trainers, and software practitioners as the athletes. They must find the time to practice, to learn the skills, and to strive towards repeatability. By showing up to practice and learning key process techniques, these teams will be the winners at acquiring, developing, and sustaining software. By embracing and institutionalizing software best practices, they will continue to be the winners. And they will be tough competitors in the software arena. Won't they also be the ones sporting a Level 5 patch on their uniforms?

In this month's issue, several articles convey the importance of process improvement and institutionalizing software best practices. In keeping with our mission, we invite you to also share practices that have paid off for your team or organization in a future *CrossTalk* issue. Send articles or comments to us at stsc.custserv@hill.af.mil. Also, if you need assistance with learning best practices, the Software Technology Support Center is here to help — see this month's insert. And remember to practice, practice, practice. ◆

CrossTalk Would Like to Ask You a Question

What was the best and/or worst software technology innovation of the 20th century?

Respond in writing at:
fax: 801-777-8069
e-mail: stsc.custserv@hill.af.mil
mail: *CrossTalk*
OO-ALC/TISE
7278 Fourth Street
Hill AFB, UT 84056-5205

We want to hear from you, so if for some reason your e-mail response did not go through, please re-send to the above address or contact us by fax or post. We will print your responses in our special December issue dedicated to the Evolution of Software Technology.

On the cover: Salt Lake graphic artist Shannon W. Ison illustrates this month's software best practices theme through an old-fashioned rendition of two boys practicing baseball skills, one wearing their idol's jersey.

Ison specializes in airbrush and computer illustration. He was the first artist to be commissioned to create limited edition Winter Olympic skis and snowboards and for the past 10 years has been the art director at a ski company in Salt Lake city where he has created one-of-a-kind handpainted skis and snowboards.

A Comment on "Applying Management Reserve to Software Project Management"

First I think *CrossTalk* is great! I just wish I would take more time to read it consistently. Keep up the great work.

Comments in reference to an article written by Walter H. Lipke in *CrossTalk* March 1999: My experience with software projects in trouble with schedule is that adjusting overtime or adding more employees or realigning employees to supposedly increase efficiency has never fixed any software schedule slippage. These were the options recommended by the author. My experience has been that the project just slips to the right further when these options are exercised. Reducing performance requirements and/or negotiating additional schedule is the best way to give the project a chance to be completed to the new schedule or requirements. The other options, in my opinion, only perpetuate a culture that needs serious improvement.

Paul Genskow
Defense Logistics Agency

The Facts about SEI's CMM-Based Appraisal for Internal Process Improvement

We would like to provide some additional information relative to "The Journey to CMM Level 5: A Time Line," an article written by Pat Cosgriff in *CrossTalk* May 1999.

The Software Engineering Institute (SEI) does not have an appraisal or assessment called a "Delta Appraisal." The SEI's CMM®-Based Appraisal for Internal Process Improvement (CBA IPI) represents an investigation at a single point in time of (1) projects defined to be within the assessment's organizational scope, and (2) key process areas (KPAs) within the assessment's CMM scope.

After sufficient data is collected during an assessment, rating may proceed for each goal within each KPA. For a KPA to be satisfied, all of its goals must be satisfied. For all KPAs within the assessment scope, the entire KPA — including all of its goals — must be investigated.

If a maturity level is desired by the assessment sponsor, all of the KPAs of a particular maturity level must be investigated, as well as all of the KPAs in lower maturity levels. For example, if an organization desires to achieve a maturity Level 3, all of the maturity Level 2 and maturity Level 3 KPAs must be investigated by collecting data for each key practice of each KPA.

Partial assessments (e.g., where some but not all goals of a KPA were rated) may be valuable as an interim activity for organizations to monitor their process improvement progress; however, such an assessment would not be considered to meet the minimum requirements of a CBA IPI.

Feedback from the community has strongly advised against partial assessments due to the opportunities for confusion or misuse. A CBA IPI must be a full assessment, examining all of the key process areas within the assessment scope, including each of the goals of each KPA, during the on-site period. It is recommended that the on-site period be completed within at most a four-week period. A CBA IPI on-site period is typically completed within a five-to-10-day period of time.

In March 1995, at the time of the Ogden assessment referenced in the above article, the CBA IPI method was in prototype stage using CBA IPI v0.3 materials. As a result of field exercises and community feedback, the method was significantly revised. CBA IPI v1.0 was released in May 1995, and CBA IPI v1.1 was released in March 1996. Although partial assessments were used experimentally in CBA IPI v0.3, there is no provision for a partial assessment to be a tailoring option in CBA IPI v1.0 or v1.1.

In the March 1995 assessment at Ogden, the team reviewed in detail the practices that they had investigated in great depth a few months earlier to determine that they were still in place. Consequently, the assessment team felt that they had done a thorough job and the results were valid.

The issue that was raised most frequently was: have they been doing these new or updated processes for sufficiently long that we could consider them to be institutionalized? There were debates during the assessment, and the team came to consensus as the results indicated.

However, in order to avoid any compromise to the integrity of an assessment, it is recommended that additional days of work be undertaken to perform a full CBA IPI.

We appreciate the opportunity to provide additional information on the subject of partial assessments.

Donna K. Dunaway, Ph.D.
Team Leader, CMM-Based Appraisal for
Internal Process Improvement (CBA IPI)

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

Correction: *On page 29 of my July 1999 article, "Confusing Process and Product: Why the Quality is not There Yet," the reference to Alan Davis in the first paragraph, last sentence should read "Alan Davis, in his excellent book* 201 Principles of Software Development, *says that good management motivates people to do their best. Poor management demotivates people [11]."*

The sentence was quite incorrect, and stated the exact opposite of what Dr. Davis said. I regret the error, and apologize to Alan Davis.

I just wish there was some way I could blame this on the Y2K problem.

— David Cook

High-Leverage Best Practices – What Hot Companies are Doing to Stay Ahead and How DoD Programs Can Benefit

Norm Brown
Software Program Managers Network

Performance-based management based on 16 industrial-strength, commercial best practices is now within the reach of all software intensive programs. Program managers in government and industry now have a choice: whether or not to utilize these commercial best practices in controlling and managing their large-scale software-intensive projects. Commercial companies' survival is critically dependent on developing and fielding software in a predictable and disciplined manner. Performance-based management is critical to their success, and such management is typically dependent on effective deployment of a small set of 16 Critical Software Practices™ that address three key areas; product integrity, product construction, and project management. The Software Program Managers Network (SPMN) has captured and described these practices for potential use by defense programs.

Cutting-Edge Companies vs. Potemkin Organizations

Extensive benchmarking of commercial and industrial large-scale software development projects leads to a remarkable conclusion: commercial companies are employing high-leverage practices most Department of Defense (DoD) projects do not even think about. Although some of these commercial projects rival the exceedingly high size and complexity of many DoD projects, most do not. The use of these practices becomes even more important and critical as project size and complexity increase.

In the course of the Software Acquisition Best Practices Initiative, the SPMN conducted, and still conducts, extensive industry best practice benchmarking. SPMN's benchmarking revealed that successful commercial companies are bottom-line driven and focus improvement activities on the big cost and schedule drivers. They are continually looking to identify specific ways to make dramatic improvements and to track their improvement progress in these high leverage areas. Can anyone imagine the Chrysler Corporation not knowing its scrap rate, not knowing the principal cost drivers for automobile manufacturing, or worse, knowing that it has a 42 percent scrap rate but doing nothing about it?

Cutting-edge companies, like Motorola's Cellular Infrastructure Division in Cork, Ireland, have figured it out. These companies use best practices to control and manage their software development, not because they are particularly concerned about cost, but because they are obsessed with minimizing time-to-market — success in the software business. These practices also reduce cost as a byproduct of reducing development time.

Cutting-edge companies have learned to identify what drives bottom-line issues of schedule, cost, predictability, and customer satisfaction, and how to drive them in the desired direction. Over the past five years, the SPMN, an Army/Navy/Air Force software development support

SPMN Best Practices Background

The Department of Defense (DoD) has, for the last few years, been stressing the importance of deploying and leveraging commercial best practices in defense programs. In furtherance of this policy, the DoD in 1994 initiated the Software Acquisition Best Practices Initiative, and asked the SPMN to lead it. The Initiative's mission, built on work the SPMN had already accomplished, was to identify and convey these software acquisition best practices to service programs.

Among the Initiative's outputs was a set of nine fundamental best practices, that were recommended to the service acquisition executives in a memo from Noel Longuemare, Under Secretary of Defense (Acquisition and Technology) and Emmett Paige, Assistant Secretary of Defense (Command, Control, Communications, and Intelligence) in 1995*.

That memo also provided for continuing best practice benchmarking and identification, and for support to programs desiring assistance in implementing these practices. Four years of continuing best practice research efforts have led the SPMN to integrate the nine fundamental practices with other essential best practices into a robust synergistic set of 16 critical practices.

The SPMN supports more than 200 programs, many of which express a fundamental need to improve how software is developed. As part of that support, SPMN consultants work with both the service program offices and their development contractors.

*Memorandum for Secretaries of the Military Departments Attn: Service Acquisition Executives, Vice Chairman, Joint Chiefs of Staff, Software Acquisition Best Practices Initiative, July 8, 1994.

organization, has benchmarked successful and not-so-successful software development activities across industry sectors in its quest to catalogue what does and does not work in the real world. The schedule and financial consequences of effectively using best practices should be obvious areas of attention for DoD software program managers.

Who is Doing What

For companies that develop software in the commercial sector, being first in the market says a lot about who will and will not survive in the global economy. It is not surprising that the commercial sector leads the way to adopting and effectively using best practices.

Motorola Iridium, in Scottsdale, Ariz., used best practices to cut defects by a factor of three, cut test time by a factor of four, and reduce overall development time by a factor of 10 — all while the software it was building grew in complexity by a factor of three. Although Motorola may have challenges with Iridium, these problems are not of the software variety. World-class companies cannot afford to allow their competitors to beat them to the marketplace. Survival of the fittest is the first law of the market.

In the defense sector, where about $42 billion [1] is spent annually on software development and maintenance, there are serious financial disincentives to cutting cost and schedule:

- lower revenue stream
- lower profit
- reduced bonuses

In the past, the defense sector's view of the industry was the more waste you produce, the more you are paid, and that includes more profit. The only reason to adopt best practices in this sector is when the market is shrinking, as it is today, but the share going to software has substantially increased, not only as a percent of the total but in absolute dollars. For example, Raytheon's Electronic Systems Division, in Sudbury, Mass., cut rework by 81 percent, tripled productivity, and substantially increased predictability.

Best practices are a way of life at Lockheed Martin where implementation of best practices is targeted to save the corporation $2.6 billion by 2003. At Lockheed Martin's Ocean, Radar, and Sensor System, productivity has increased by a factor of six and errors have been reduced by a factor of 25 through the implementation of best practices.

Boeing is another success story that benefits from best practices. They carefully monitor their rework metrics and rework drivers, and have managed to bring their rework substantially below industry norms. That translates to substantially improved time-to-fielding and associated cost savings.

Dark Clouds on the Horizon for Software Waste

What will it take to make the rest of the defense sector take notice of the rampant waste that exists in software development? In the commercial sector, it is not inconceivable that stockholder derivative suits will be the driving influence on medium and large companies that develop software to support their operational mission and not as a commercial endeavor to adopt best practices. These suits will put corporate directors on notice as the ones responsible for excessive software development costs.

The congressional sector has become active on the taxpayer's behalf, and has become concerned and quite interested in programs using best practices. The House Armed Services Committee in its year 2000 defense bill directed, "mandate[ing] the use of identified best practices for software development and management for all acquisition programs" [2]. The Senate Armed Services Committee (SASC) in its year 2000 defense bill requested a report from each of the services on best practices implementation [3]. SASC is concerned that "…not enough has been done to adopt management best practices to the acquisition, development, and maintenance of software defense-wide" [3]. The SASC report requests that "…the [Defense] Department report to Congress by February 1, 2000 on its efforts to identify and adopt best practices in software development." It also requires including six specific metrics in the report.

If the department implements this direction, great improvement in cost and schedule will come to defense projects and to the entire defense industry.

A Fundamental Difference in Approach: Practices vs. Processes

Best practices and process improvement aim to achieve improvements in how software is developed. While process improvement serves as a floodlight on what can be done, offering a rich spectrum to choose from, best practices are the laser beam, pinpointing high-leverage activities directly coupled to the bottom-line. The bottom-line is improved by focusing on related implementation detail. The focus should not be merely on generic process improvement, but on what really counts. And what really counts are the underlying cost and schedule drivers — critical best practices that attack these cost and schedule drivers.

An important aspect of critical best practices is that they can be immediately applied. They are independent of the Software Engineering Institute (SEI) maturity level an organization may be at today — although you will have to tailor the practices to the circumstances of your specific project[1]. Another aspect of critical best practices for organizations interested in moving-up the SEI Capability Maturity Model® (CMM) ladder is that these practices constitute a set of tactical disciplines that move organizations about 80 percent of the way to CMM Level 3, and a rapid, early implementation of high-payoff practices. The practices provide the tactical detail that the strategically oriented CMM does not address. Consultants in CMM improvement have indicated that SPMN's 16-Point Plan is an effective template for substantially reducing the 18-24 months it typically takes to create a CMM improvement plan from scratch.

Three Areas of Attack

The 16 Critical Software Practices have emerged as a product of the Airlie Software Council from its initial work in 1994-95 in identifying nine essential best practices. These form the core of the 16 Critical Best Practices, augmented with additional understanding of commercial practices by continued benchmarking and

"High Leverage Best Practices..." continued on page 14.

16 Critical Software Practices for Performance-Based Management

Jane T. Lochner
U.S. Navy

The 16-Point Plan™ focuses on effective management and technical processes for improving the bottom-line: detecting defects, managing complexity, reducing rework, eliminating excessive and unnecessary costs, and increasing productivity. It addresses three primary areas of software management: project control, product construction, and product integrity. The practices were forged in the crucible of real-world pressure to succeed and represent the combined experience of successful program managers and industry leaders. Recognizing that change is difficult, the 16-Point Plan recommends small but powerful steps that can be introduced into an established program.

THIS ARTICLE DRAWS ON information in the Software Program Managers Network's (SPMN) new program manager's guide titled, *16 Critical Software Practices for Performance-Based Management*, which is due for initial release in Fall 1999. The 16 Critical Software Practices™ for Performance-Based Management (the 16-Point Plan) are applicable to all large-scale, software-intensive projects (i.e. projects relying on the full-time efforts of 12 or more people). The practices that comprise it, however, are scaleable, all or in part, to smaller projects and to different software project environments. The 16-Point Plan was developed by SPMN with assistance from members of the Airlie Software Council, about 20 of the nation's leading software experts convened to assist SPMN in the identifying of industry best practices. The guiding principles used in developing each of the practices were that each practice be:

- applicable to all types of software and life cycle models
- flexible
- nonproprietary
- specific
- measurable
- realistic and attainable
- readily implementable

The practices identified satisfy these guidelines, making the 16-Point Plan a firm foundation for project success. This plan should be used according to the circumstances and environment of a given project, including where it is in its life cycle when the 16-Point Plan is first adopted. All practices are generally applicable to both government and industry projects and to nearly all domains. The plan is focused on effective management and technical processes, including techniques for finding defects as they occur, managing complexity, reducing rework, eliminating excessive and unnecessary costs, increasing productivity, and other beneficial effects.

The practices that comprise the 16-Point Plan are termed "critical" because software project managers and organizations, whose bottom-line performance is consistently better than average, use these practices and consider them essential. The bottom-line that the buyers of software development are interested in consists of:

- end-user satisfaction
- development and maintenance cost
- time-to-market
- quality
- predictability of final cost and schedule

Each of these critical practices is supported by metrics from past large-software-intensive system development and maintenance projects. These practices have been forged in the crucible of real-world pressure to succeed.

The 16-Point Plan is not presumed to be an exhaustive set of practices. However, the plan represents the combined experience of successful program managers and industry leaders. It will go a long way toward engendering success in any software development or maintenance effort.

As illustrated in Figure 1, the 16-Point Plan addresses three primary areas of software management:

- Project Control. It includes those practices that result in the identification of basic project constraints, expectations, and metrics. It also encompasses practices to plan and implement a project environment to predictably satisfy customer expectations and constraints.
- Product Construction. Includes those activities that specify the basic product requirements; maintain traceability to these basic requirements; and control content, change, and use of the many artifacts and deliverable products that are produced to satisfy user and customer requirements and expectations.
- Product Integrity. This ensures that defects, which occur as part of the software process, are identified and removed in a timely fashion. Product integrity ensures that testing is complete and effective and results in the right product consistent with agreed-to requirements and actual expectations.

While the practices that comprise the 16-Point Plan are individually useful, their complementary nature provides a strong synergistic effect when used as an integrated set. Using them will not guarantee success, but they can help facilitate it.

Those familiar with process improvement models, such as the Capability Maturity Model (CMM®), will quickly realize

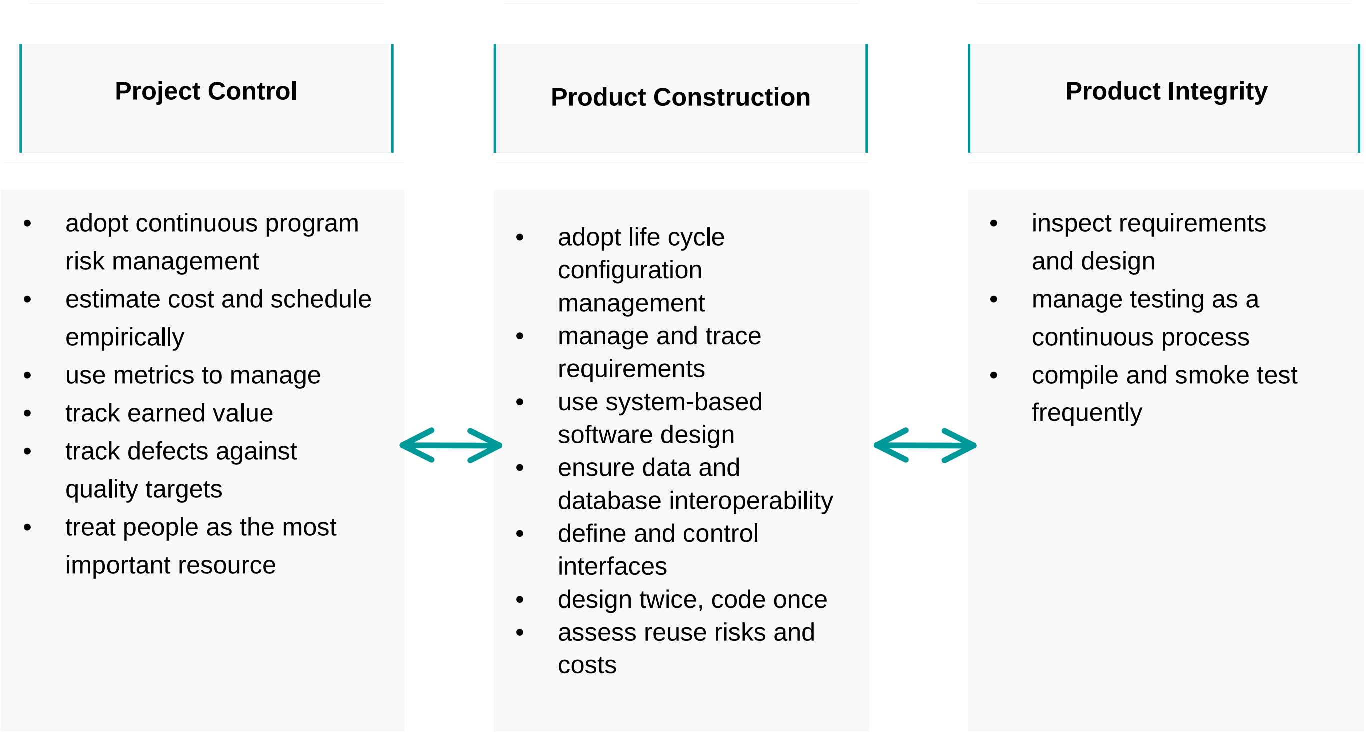

Project Control
Product Construction
Product Integrity
• adopt continuous program risk management
• estimate cost and schedule empirically
• use metrics to manage
• track earned value
• track defects against quality targets
• treat people as the most important resource
• adopt life cycle configuration management
• manage and trace requirements
• use system-based software design
• ensure data and database interoperability
• define and control interfaces
• design twice, code once
• assess reuse risks and costs
• inspect requirements and design
• manage testing as a continuous process
• compile and smoke test frequently

Figure 1. *The 16-Point Plan.*

that these practices supply tactical solutions to the model's strategic orientation. The practices map to many of the model's key process areas and should assist organizations striving to advance to the next CMM maturity level.

These practices are straightforward, readily implementable techniques. Although some practices may require training in basic skills, such as conducting structured meetings as a necessary foundation for formal inspections, they can, for the most part, be implemented without making investments in new equipment, technologies, or staff. Cultural resistance to the discipline inherent in these practices and to the management visibility that comes from several of the practices, is the biggest obstacle to successfully implementing these practices. Another stumbling block is that organizations inexperienced in some of the practices in the plan may think they are unnecessary. Because they do not use the practice at the present time, they may not recognize the value and benefits to their organizations that implementing these practices would bring.

The 16-Point Plan can be used with an established program. It recognizes that change is difficult and recommends small but powerful steps to initiate each of the 16 practices. Although the 16 practices cannot make successful those programs that are inadequately funded, without proper staffing, and faced with impossible schedule deadlines, implementing these practices can minimize damage.

The 16-Point Plan

The 16-Point Plan takes a two-dimensional approach to the critical practices. First, it takes a vertical approach, explaining the details of each practice. Each element is examined, identifying "practice fundamentals." These are key principles that outline the essence of the practice. Implementation guidelines also are identified. These are practical steps that can help implement the practice in a given program. A set of "quick look" questions is provided to help the program manager make a crude assessment of whether his/her program has a potential problem in each area and a list of "alarms," which relate to lessons learned in each area. Each practice specifies associated metrics that the project manager should monitor. Finally, each practice concludes with a list of detailed questions that should be asked if a project is unable to satisfactorily answer the "quick look" self-assessment questions, following with the recommended corrective actions.

Then it takes a horizontal perspective, describing how individual practices might be sewn together into an effective program. The 16-Point Plan associates the 16 critical practices by program phase. Taken together, these two perspectives provide a model that can be applied to any project and move it toward success.

The following is an example of some of the practice details contained in the 16-Point Plan.

Project Control

To be successful, the right project environment must be established. This environment must be conducive to establishing a stable team, identifying, organizing, and coordinating tasks, and recognizing and mitigating risks that can impede success. Most of all, this environment must make it as easy as possible for team members to pull together toward success.

Project control brings together a series of six management practices that can help establish a success-oriented environment. These provide early indicators of potential problems, coordinate

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

the work and the communications of the development team, and achieve a stable development team with the needed skills. These practices are essential to delivering the complete product on time, within budget, and with all documentation required to maintain the product after delivery.

Achieving project control requires developing a detailed activity network for all effort to at least the next delivery, an estimate of the cost and schedule for this effort, and allocation of the cost and schedule estimate. Planning is the basis of project control, establishing a method for conducting business, a management process, and a quantitative basis for monitoring progress and risk. The goal of planning is to establish a working project management environment, not the production of a plan for the sake of meeting a requirement to produce a plan. "Use Metrics to Manage" is one project control critical practice. Examples of "quick look" self-assessment questions that might be asked regarding metrics are:

- Have threshold values been established?
- When was the last time the metrics showed an anomaly (were not what was expected)?

Alarms that indicate that the metrics program is not being taken seriously include:

- A large price tag attached to request for metrics data.
- Rebaselining is frequently required.

Collecting metrics solely to collect metrics is not a best practice. It is, in fact, counterproductive. Since the value from a metrics program comes from the actions taken as a result of metrics analysis, one should track the percentage of decisions made based on metric data.

When the "quick look" self-assessment questions indicate a lackadaisical metrics program, the program manager should asked more probing questions such as:

- Are there threshold values for early problem indication metrics that trigger reporting to higher levels of management? If so, for each metric with such threshold values, what are the threshold values and to what level of management does each value trigger a report?

A worthwhile metrics program must measure the right metrics. The following steps can be used to identify the proper metrics:

- Define program issues/problems/risks
- Identify reporting obligations/needs
- Determine what indicators would show problem areas
- Sort indicators into metric categories
- Determine the delta from metrics currently collected and metrics needed
- Identify additional reporting collection delta
- Identify ranges/metrics

Product Construction

Projects need a common means of doing business as well as a common language process during construction to ensure communication among suppliers/developers, users, programmers, analysts, project leaders, program managers, and the program executive officer.

Although projects are never the same, the process should be consistent because projects require discipline and predictability. Before planning how something will be accomplished, it is useful to understand what has to be done. Techniques must be defined before they can be integrated into a project; and while innovative technology is often required to meet project goals, their impact must be realistic and have broad project support. Esoteric solutions and the use of leading edge technology not tailored to project objectives are counterproductive.

Essential to construction are tools that sustain project requirements, not vice versa. Automated tools, supported by configuration management, solve project problems more efficiently than manual techniques. However, automated aids are useful only if they satisfy an identified need and are defined and selected in a top-down sequence. All too often construction starts with the tool, forcing the tool to fit the problem. Improper tool selection and application result in data rework at the low end and wasted work/scrap at the high end.

The discipline of configuration management (CM) is vital to the success of any software development effort. Two questions which give a CM "quick look" are:

- Can you access the earliest/most recent version of a software system?
- Can you produce the change documentation for the approved last change to the current system?

A CM process is probably not effective if any of the following alarms occur:

- The Configuration Control Board (CCB) merely rubber stamps requests; requests are submitted "after the fact." There is not a mixture of accepted, rejected, and held for further investigation actions.
- The CM process is considered level-of-effort and not tied to specific tasks/products.

Some metrics which measure the effectiveness of CM are:

- number of days since last change to library documents
- turn-around time for CM products

When uncertainty arises concerning the CM process, one should ask detailed questions such as:

- Have several people described the CM approach and process? Are these descriptions consistent? Do they match the documented process?
- When under heavy schedule pressure, are changes made to code without going through a controlled change process managed by CM?
- Are CCBs fully assessing the impacts of each proposed change or the risk and cost of making the proposed change prior to authorizing that change? Are all impacted configuration items identified?

For CM to be effective it must be empowered. Corrective actions to empower the CM team include:

- Charter the CM organization. Give it a clear mission, responsibilities, and authority.
- Staff it adequately with experienced developers.
- Train the team thoroughly in the tools that are to be used.

Product Integrity

Software development is a continuum of events, one building on the next. If one is done poorly, subsequent activities that build on the work suffer. Project success and acceptability criteria depend on managing the project to ensure quality. Generally, there will never be time to clean it up. When problems occur, options are limited. It is better to manage quality from the beginning.

"Compile and smoke-test frequently" helps ensure that the product is growing in a controlled manner. These questions provide a "quick look" assessment of how well an organization is following this practice.

- Is the build of the current system baseline more than five days old?
- Can the CM group build the current system baseline unaided?

Management needs to scrutinize the compile and smoke-test process if any of the following alarms occur:

- gradual increases in the number of changes included in builds or in the time between builds
- use of binary patches

Management can use the following measures to monitor the compile and smoke-test practice.

- days since last build
- number of problems identified during smoke-tests

When the compile and smoke-test practice needs redirection, detailed questions, such as the following, can help pinpoint problem areas.

- Can the customer explain the regression and smoke-test philosophy to an outside organization?
- Do the regression and smoke-test suites address all capabilities in the current configuration?

Implementing an effective smoke-test strategy requires:

- building systems and executing tests at least twice a week.
- smoke-testing systems built only from the central CM library. Test files, stubs, drivers, or other components not held by the CM system must not be used.
- smoke-tests based on a pre-approved, traceable procedure run by an independent organization — not the engineers who produced the change package.

Conclusion

The 16-Point Plan integrates the critical software practices into a road map that can help program managers navigate around the hazards and obstacles that often block the path to success. It is a set of high-leverage practices that distill the experience of successful program managers into an executable strategy that can be applied to virtually any development effort. It is a starting point for structuring and deploying an effective process for managing large-scale software development and maintenance, but must be tailored to the particular culture, environment, and phases of a program.

The 16-Point Plan incorporates proven commercial best practices and focuses on the essential details of each practice necessary to achieve high return on investment, bottom-line improvements. Together the practices constitute a powerful set of technical and management disciplines that can be put in place quickly to achieve rapid bottom-line results. Successful implementation of these essential details should ensure big savings. Of course, these practices cannot save "death march" programs that are expected to deliver under impossible schedule deadlines with inadequate funding and without the required skilled staff. ◆

About the Author

Jane T. Lochner is a 1984 U.S. Naval Academy graduate. She served aboard USS Norton Sound (AVM-1) and USS Cape Cod (AD-43). She was selected to the Engineering Duty community in 1988. She has extensive experience with developing and fielding complex, real-time combat systems on aircraft carriers and large-deck amphibious ships. Currently, she is assigned to the Office of the Assistant Secretary of the Navy for Research, Development, and Acquisition working command, control, communications, computers, intelligence, surveillance, and reconnaissance and interoperability issues. She holds a bachelor's degree in marine engineering, master's degrees in logistics, applied physics, and computer science, and is a graduate of the Defense Systems Management College Program Manager's course.

U.S. Navy
1000 Navy Pentagon
Washington, D.C. 20350-1000
Voice: 703-602-6887
Fax: 703-601-0346
E-mail: lochner.jane@hq.navy.mil

Software Mini-Assessments: Process and Practice

Gary Natwick, Geoff Draper, and Lennis Bearden
Harris Corp.

This article describes a software mini-assessment rating and evaluation method that provides projects with a quick and easy way to assess its software process maturity, and provides organizational insight into the success of software process institutionalization efforts.

Harris Information Systems Division (HISD) achieved the Software Engineering Institute's (SEI) Software Capability Maturity Model (SW-CMM^{SM}) [1] Level 3 in 1994 and is pursuing Level 4. As part of the effort to maintain and advance this process maturity, the division's Software Engineering Process Group (SEPG) performs periodic software mini-assessments on key projects to determine the strengths and weaknesses of project and organizational processes. The mini-assessment is used as a baseline against which to assess progress toward division goals such as improving process maturity, adherence to division standard software processes, and institutionalization of technology insertion efforts.

Method Overview

The assessment method is based on a CMM-based progress assessment process proposed by Michael Daskalantonakis of Motorola [2] and SEI CMM-based Appraisal for Internal Process Improvement (CBA IPI) vol.1 [3, 4]. In this method, the key activities of each CMM key process area (KPA) are rated according to scoring guidelines in each of the following dimensions:

- Approach — reflects the organizational commitment and managerial support for the practice, as well as organizational capability to implement the practice.
- Deployment — an indicator of institutionalization, reflecting the breadth and consistency of practice implementation.
- Results — assesses the effectiveness of the practice and its positive results over time.

Table 1 provides a matrix of scoring guidelines to be applied to the KPA activities, to ensure the spirit and themes of the CMM are addressed. Each KPA activity is rated 0-10 in each of the dimensions, with each dimension equally weighted. An odd-numbered score is possible if some, but not all, of the criteria for the next higher level have been met. Scores in each dimension are averaged to generate a summary score for the KPA activity component. The KPA goals (described in the CMM) are also scored, based on the individual scores of each activity that maps to the goal. Evidence of project artifacts (e.g. documentation) is noted to substantiate adoption of the key practice.

Overall scores for each KPA are obtained by averaging the scores for each component goal and activity, indicating how well the KPA practices have been implemented within the organization. In general, a score of seven or above indicates a satisfactory score, and likelihood that the KPA will be judged acceptable in a formal capability assessment or evaluation. Low scores identify improvement needs for key activities and KPAs necessary to raise the organizational process maturity. In addition, the KPA scores for a given CMM maturity level can be reviewed to determine an overall summary CMM rating; all KPAs must be rated with scores of seven or above in order to be assessed at a given maturity level. For example, all six Level 2 KPAs and all seven Level 3 KPAs must be rated at least seven or higher in order to obtain an overall rating of Level 3.

To account for satisfaction of overall CMM KPA goals, via a mapping of key activities to goals, the HISD Engineering Process Group (EPG) enhanced the Motorola method. Satisfaction of every KPA goal is a critical CBA-IPI factor in determining satisfaction of the KPA. A history mechanism has also been added to reflect progress since the last assessment. An Excel spreadsheet has been developed to capture assessment data and automatically generate reports for CMM compliance. In addition, a cross-reference to division processes and CMM-specified evidence has been added to every KPA activity to assure completeness in the understanding and response to each evaluation score.

Application

The progress assessment method is used by the HISD EPG primarily to assess the process maturity of individual programs. Currently, only the software process maturity is assessed; however, this method will later extend to support assessments of other functions, such as hardware and system engineering. HISD has adopted the CMM framework for systems engineering process maturity, and has internally developed a hardware CMM process maturity framework for hardware engineering. HISD sponsors separate process teams for each of these disciplines as owners of their respective engineering processes, with the EPG as the integrating process development, management, and improvement mechanism.

To conduct mini-assessments, the EPG holds a kickoff meeting for the mini-assessment participants, who separately complete their ratings of CMM key activities. The EPG collects and consolidates individual ratings, and facilitates a consensus meeting at which final ratings for each key activity are achieved. We have found these consensus meetings to be one of the most valuable parts of the mini-assessment process, as project members across multiple functional disciplines focus on and discuss their project processes in what tends to be almost a team-building exercise. The SEPG analyzes resulting scores and gener-

The Software Capability Maturity Model (SW-CMM) is a service mark of Carnegie Mellon University.

Peer Reviews Goals

1. Peer review activities are planned.
2. Defects in the software work products are identified and removed.

Score	Key Activity Evaluation Dimensions		
	Approach	**Deployment**	**Results**
None (0)	• No management recognition of need • No organization* commitment • Practice not evident	• No part of the organization* uses the practice • No part of the organization* shows interest	• Ineffective
Poor (2)	• Management has begun to recognize the need • Support items for the practice start to be created • A few parts of the organization* are able to implement the practice	• Fragmented use • Inconsistent use • Deployed in some parts of the organization* • Limited monitoring /verification of use	• Spotty results • Inconsistent results • Some evidence of effectiveness for some parts of the organization*
Weak (4)	• Wide but not complete commitment by management • Road map for practice implementation defined • Several supporting items for the practice in place	• Less fragmented use • Some consistency of use • Deployed in some major parts of the organization • Monitoring/verification of use for several parts of the organization*	• Consistent and positive results for several parts of the organization* • Inconsistent results for other parts of the organization*
Marginal (6)	• Some management commitment; some management becomes proactive • Practice implementation well under way across parts of the organization* • Supporting items in place	• Deployed in some parts of the organization* • Mostly consistent use across many parts of the organization* • Monitoring /verification of use for almost all parts of the organization*	• Positive measurable results in most parts of the organization* • Consistently positive results over time across many parts of the organization*
Qualified (8)	• Total management commitment • Majority of management is proactive • Practice established as an integral part of the process • Supporting items encourage and facilitate the use of the practice	• Deployed in almost all parts of the organization* • Consistent use across almost all parts of the organization* • Monitoring /verification for almost all parts of the organization*	• Positive measurable results in almost all parts of the organization* • Consistently positive results over time across almost all parts of the organization*
Outstanding (10)	• Management provides zealous leadership and commitment • Organizational excellence in the practice recognized even outside the organization*	• Pervasive and consistent deployment across all parts of the organization* • Consistent use across all parts of the organization* • Monitoring /verification for all parts of the organization*	• Requirements exceeded • Consistently world-class results • Counsel sought by others

* Evaluations can be performed for single components, programs, or organizations, as applicable.

Notes:

Each KPA key activity is rated separately in each of the above dimensions.
Odd-numbered scores can be assigned if some but not all of the criteria are satisfied for next higher level.
Dimensions are equally weighted to develop overall score for each key activity.
Key activity scores are rolled up into scores for KPAs and overall CMM Level.

Table 1. *Guidelines to rate CMM Key Activities.*

ates a report for briefing of assessment results back to the project.

The process is intended to minimize impact to the program and its staff, yet provide a meaningful assessment of program process strengths and weaknesses for continuous improvement. No inspection of program evidence is performed; however, mini-assessment worksheets provide for recording pointers to such evidence should it be necessary to collect it for a formal process assessment or evaluation. The EPG provides automated tool support (Excel spreadsheet) to support entry, tabulation, and reporting (graphs) of scoring ratings. Organizations may obtain a soft copy of this spreadsheet by contacting the authors via e-mail.

Process Steps

The general steps necessary to deploy the assessment are described below. The estimated project staff time is two hours for debrief/feedback.

1. Select project to be assessed, and determine participants. The EPG schedules at least one mini-assessment every other month, with programs selected on a rotating basis to ensure coverage of different product lines and project types (e.g. new development, operations and maintenance, internal research and development). Mandatory program participants for the software assessment include the project manager, chief software engineer, and software quality assurance, at a minimum. Recommended participants include cognizant software engineering managers, program subsystem leads, program management, systems engineering, and other functional representatives on either a full-time or as-needed basis (e.g. configuration management or subcontract managers for applicable CMM

KPAs).

2. Brief participants. The EPG assessment team convenes an overview meeting to describe the assessment goals and methods.
3. Participant preparation. The individual participants review the KPA goals and activities, and prepare their own notes and ratings in advance of the assessment utilizing the SEI CMM KPA worksheets provided by the EPG. Examples of the worksheet for a single KPA are shown in Figure 1. Substantiating evidence or examples of applying the practice should be recorded on the forms, but are not physically collected. Entries in the soft copy spreadsheet can be e-mailed to the facilitator to help streamline the assessment meeting.
4. Conduct assessment. The assessment team and participants convene a meeting at which all individual rating inputs are discussed and consolidated using a Delphi technique to converge on a consensus score on each of the dimensions (approach, deployment, and results) for each KPA activity. Where consensus can not be reached, the lowest score is used. The assessment team facilitates and guides the discussions, questioning the participants on each activity and its process artifacts. The results are recorded in hard copy for subsequent transcription to soft copy media.
5. Consolidate results. The assessment team enters the evaluation scores into a spreadsheet, which consolidates and reports the composite results. A summary chart is generated, an example of which is shown in Figure 2, that depicts the assessed summary rating of each KPA, in addition to the range of scores of the strongest and weakest KPA goals. The KPA goal scores are determined by the scores of the individual key activities mapped to each goal, as described in the CBA IPI Lead Assessor's Guide [4]. The assessment team develops a summary briefing of assessed strengths and weaknesses, and recommendations for areas needing improvement.
6. Review results. The assessment team and participants reconvene to review the assessment findings, including strengths, weaknesses, and recommendations for improvement.
7. Develop action plan. The assessed organization develops an action plan to address weaknesses identified during the assessment. Actions may be assigned to the program or to the division EPG. Action plans for program weaknesses are typically internal to the program, unless division interests are directly jeopardized (e.g. severe weakness that would impact division assessment level).
8. Follow-up. The organization and/or EPG tracks the actions called for by the action plan and monitors the implementation status.
9. Process improvement. On a regular basis (at least annually), the EPG analyzes the process maturity progress of the organization across all projects and reports the results to senior management. The report focuses on the overall organization, not individual projects.

Benefits

The consolidated mini-assessment scores and findings across the division help identify both opportunities for process improvement, and project strengths that may be beneficial for wider adoption across the division. In the past two years, HISD has performed the mini-assessment method on nine projects. The analysis of these findings (in addition to other sources such as risk assessments, process improvement requests, and customer evaluations) have been key to targeting investments in strategic improvements. Because the mini-assessments are performed on projects selected to provide a cross-section of the division, they provide a good way to assess the institutionalization of defined Level 3 processes, which will be a crucial element in the pursuit of Level 4. As such, the mini-assessment findings are closely scrutinized by the EPG and all levels of engineering management, up to and including division senior management. Action plans are generated and tracked to ensure progress on assessed weaknesses, and toward strategic division business goals. Some of

Figure 1. *Sample mini-assessment worksheet for peer reviews KPA.*

Key Activities				Evaluation Dimensions					Goal Mapping	
Number	Description	Division References	CMM Specified Evidence	Approach	Deployment	Results	Score	Project Artifacts/Notes	1	2
1	Peer reviews are planned, and the plans are documented.	Peer Review Handbook	plans						*	
2	Peer reviews are performed according to documented procedure.	Peer Review Handbook	documented procedure							*
3	Data on the conduct and results of the peer reviews are recorded.	Peer Review Handbook 2.5 Report Results	peer review data							*

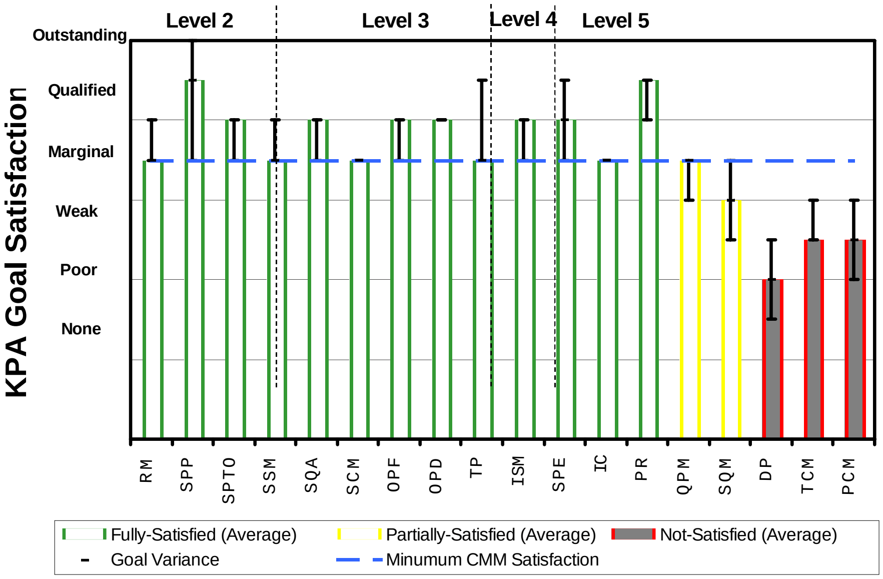


Figure 2. *Example mini-assessment summary analysis.*

the benefits realized include:

1. Project process review — The technique forces the project team to spend time reviewing their process from a SEI CMM perspective.
2. Project team building — By having project leaders across multiple disciplines focused simultaneously on process, many project problems and process integration issues have been surfaced for discussion and resolution.
3. EPG project awareness — The technique has provided the EPG with in-process feedback on the strengths and weaknesses of division and project processes.
4. Project process improvements — The technique has identified project process problems that have been addressed by project process improvement teams.
5. Division process improvements — analysis of mini-assessment data from multiple projects has identified organization process weaknesses that the EPG addressed.

In addition, the results from recent formal software capability evaluations (SCEs) show a high correlation with the findings from the mini-assessment method. The mini-assessment method is not a substitute for a formal SCE or CBA-IPI; it complements those methods by providing a quick and easy method for identifying interim process improvements. Due to the success of the tool within HISD, it is being used by several divisions across the Harris Corp. Future enhancements to the process will include:

- expansion of mini-assessments to other functional disciplines and CMM frameworks.
- addition of a method for reassessing projects.
- strengthening the closure plans to ensure all problems raised by the mini-assessment are resolved.

Conclusion

The mini-assessment process practiced by Harris Information Systems Division is a key element of our organizational process improvement strategy. It provides a low-cost but high-yield approach to assessing process maturity and compliance that has proven beneficial to the division and its projects. The authors would be pleased to support requests for additional information on the mini-assessment method, tools, or experience. ◆

About the Authors

Gary Natwick is the metrics leader for the EPG responsible for advancing the Harris Information Systems Division (HISD) to SEI SW-CMM Level 4. Previously, he was the leader of the SEPG advancing the HISD software process maturity to SEI SW-CMM Level 3. He has more than 25 years of software engineering experience (management, development, and process improvement) with Harris Corp. and the Air Force. He earned a bachelor of science degree in electrical engineering from the University of Miami. Natwick is a member of the Institute of Electrical and Electronics Engineers and the Association for Computing Machinery and is an Authorized Lead Assessor in the SEI CBA-IPI method.

Harris Corp.
P.O. Box 37
Melbourne, Fla. 32902
Voice: 407-729-3970
Fax: 407-729-3090
E-mail: gnatwick@harris.com

Geoff Draper is the software focus leader of the EPG responsible for HISD software process definition and improvement. He has more than 15 years experience with Harris Corp. in various software development and leadership positions. Draper earned a bachelor and master of science degrees in computer science from the University of Illinois and Florida Institute of Technology, respectively.

Harris Corp.
P.O. Box 98000
Melbourne, Fla. 32902
Voice: 407-984-5864
Fax: 407-984-6323
E-mail: gdraper@harris.com

Lennis Bearden was the leader of the EPG responsible for all HISD engineering process improvements. He has more than 25 years experience covering all aspects of system development, including hardware, software, system engineering, and program management. His interests are software processes, systems engineering process, and systems architecture. Bearden earned a bachelor and master of science degrees in electrical engineering from the University of Tennessee.

GE Harris Railway Electronics
407 North John Rodes Blvd
Melbourne, Fla. 32934
Voice: 407-757-8702

Fax: 407-242-4118
E-mail: lbearden@ge-harris.com

References

1. Paulk, Mark, et al., "Capability Maturity Model for Software, Version 1.1," Technical report CMU/SEI-93-TR-24, Software Engineering Institute, Pittsburgh, Pa., 1993.
2. Daskalantonakis, M.K., "Achieving Higher SEI Levels," IEEE Software, July 1994, Vol. 11, No. 4, pp. 17-24. Reprinted in *CrossTalk*, September 1994, Vol. 7, No. 9, pp. 12-18.
3. Dunaway, Donna K. and Masters, Steve, *CMM Appraisal for Internal Process Improvement (CBA IPI): Method Description* (CMU/SEI-96-TR-007). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1996.
4. Dunaway, Donna K. *CMM-Based Appraisal for Internal Process Improvement (CBA IPI) Lead Assessor's Guide* (CMU/SEI-96-HB-003). Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pa., 1996.

"High Leverage Best Practices..." continued from page 5.

project consulting experience. They have all been successfully tested in the crucible of successful large-scale software projects.

The Airlie Software Council identified three major areas of software development the 16 Critical Practices address:

- product integrity
- product construction
- project control

These areas and subsumed practices can be found developed further in this journal in Jane T. Lochner's article on page 6. The practices are useful for controlling complexity inherent in all large-scale software projects — and keeping it from spinning into uncontrollable chaos. Each practice makes a high-leverage contribution and are "high-leverage" practices because of the relatively low cost, quick implementation, and dramatic effect on the bottom line.

Where the Rubber Meets the Road

The critical practices and related implementation both defined in the 16-Point Plan were selected to deliver maximum leverage to programs wanting to dramatically improve their bottom-line and to expedite progress in organizations desirous of moving to the SEI CMM Level 3.

The CMM serves as a meaningful strategic framework for process improvement; the 16 Critical Practices constitute a tactical infrastructure that enables software development organizations to effectively address many of the CMM's Key Process Areas (KPAs).

Although these 16 Critical Practices serve this infrastructural role to CMM KPAs, their fundamental role is independent of this relationship to the CMM — they focus, at their essence, specifically on addressing improvements to the bottom-line — enabling significantly reduced time-to-field and related cost reduction and quality improvements. Although many of the CMM KPAs have similar, if not identical titles as critical process, they are largely two sides of the same coin.

This plan was devised to enable and facilitate an effective and straightforward implementation of critical best practices. Discussions with numerous consultants who assist organizations with CMM improvements make it clear that companies typically take between 18 and 24 months to design an improvement plan for getting from CMM Level 2 to Level 3; and during this time much of the process improvement momentum dissipates and management support wanes. The 16-Point Plan can serve well as a template for reaching Level 3. Since CMM Level 2 has a significant focus on improvement in project management and Level 3 has a key focus on team effectiveness, the critical best practices address both of these key improvement areas.

What You Can Do

1. First determine whether or not your project has a detailed plan of all activities needed to achieve the next milestones, together with or including the personnel resources and time allocations necessary for this completion.

Although obvious, many programs lack this detailed planning. Without it, tracking by earned value will be meaningless, schedule compression cannot be completed, critical path and near-critical path cannot be identified through statistical schedule verification, tools cannot be employed, risk identification capabilities will be diminished, and you will not be able to use schedule automated control and authorization tools. If such a detailed plan does not exist, have one made.

2. Ensure that the effective structured peer reviews trend of a Fagan-like variety are being conducted to all detailed task products; that such reviews constitute task completion criteria for earned value and configuration management purposes; and that architectures are being modeled and simulated.
3. Ensure that a "bottom-up" risk management process is in place — one that has risk identification facilitated among front-line developers with management involvement; risk mitigation planning for high impact, high probability risks that a risk officer can manage and focus the process; and a culture that rewards risk identification — not punishes it. Be sure the likelihood of key development personnel suddenly leaving the project is considered as a major risk. If the project is planning a heavy reliance on reuse, then ensure that this is noted as a major risk as well.
4. Consider the 16 Best Practices and prioritize them in accordance with the needs of your particular program.

More About Best Practices

SPMN has also developed a template plan for large-scale defense projects: the 16-Point Plan for Performance Based

"High Leverage Best Practices..." continued on page 27.

Experience in a Bottle: How Boeing Captured its Assessment Best Practices

Darrell Corbin, Russ Hamerly, and Roger Cox
The Boeing Co.
Dr. Kenneth Knight
Seattle Pacific University

How do you "bottle" the years of experience from your hot project teams? These might be an emergency review of a critical information system (IS), assessments of potential suppliers, or evaluation of IS organizations. This article describes how Boeing captured and used tailorable best practices and ended up creating a companywide Web site, the Structured Review Process (SRP).

The Age-Old Problem: Starting From Scratch

How many times have you been part of this scenario: Your boss informs you that Alpha Team has been formed to review (assess or audit) a development project for a new information system, named Critical. It has to be done in four weeks. The project is essential to the company and the new chief information officer (CIO) wants to know if Critical will be done on time. Also, the CIO wants assurance that the project is managed properly and if Critical can be used in other divisions of the company. You have been selected as the team project manager.

If you are like most other teams of this type, you get everyone together, try to decide exactly what it is you are supposed to do, then start doing it. This is an all-star team, but how you do the review is up in the air. Your team develops a new process, complete with its own deliverables. Then, like a development project, you start coding, in a manner of speaking. After all, you only have four weeks to complete the review.

If you are lucky, your organization has a standard process for conducting assessments. The process is documented, repeatable, and has all those other favorable characteristics so well described by the Software Engineering Institute in its software Capability Maturity Model (CMM®). But this is not your lucky day.

What happens when Alpha Team is done? Who takes time to document the lessons learned, collect deliverable examples, and document the process for future teams? Probably no one. After all, there is real work to be done and it has stacked up for four weeks while the team has conducted the review.

Teams, Teams Everywhere

Boeing has more than 200,000 employees and there are always dozens of teams working on important, time-critical projects. Several recent projects in Commercial Airplane Information Systems have assessed the ability of potential foreign suppliers to perform computing work for Boeing. That is how the SRP Web site got started. The general areas of work included Y2K, porting of engineering applications to new platforms, and business system maintenance.

The history of this process goes back to 1997 when a 12-member multi-discipline team formed to assess suppliers in Elbonia (the name we will use in honor of Dilbert, our favorite software engineer). The review had to be done quickly, and the sponsors left it to the team as to how to do the review. Most of the team members had never done an assessment, so they looked for help. Fortunately, some of the members were familiar with the CMM. Boeing had adopted the CMM for division software process improvement activities and had several years' experience conducting CMM assessments. The team used the general CMM approach of using a questionnaire, conducting interviews, and identifying actions. The CMM was a key source of questions for the project management and software process maturity sections of the questionnaire.

Team members completed the assessment on time, received an award, and returned to their regular jobs.

Fortunately, the team project manager carefully archived the process description developed by the team, including deliverable templates and samples, some lessons learned, and related documentation. This step was fortunate because in a few months came another hot project — assessment of foreign suppliers in Monrovia. This time the companies would not be interviewed in depth by a team of assessors, but visited by two managers for half-day reviews.

One of the managers happened to know about the Elbonia assessment and asked if its process could be tailored for the next assessment. The answer was an immediate "yes." The process was quickly modified for this scaled-down assessment and the managers flew to Monrovia and completed their review in a few short days. Reuse had started.

A few months later another hot project beckoned — review of a development project, the Dogbert system. Dogbert was extremely important because it supported a new Boeing product that was about to be released. Late products mean unhappy customers and Boeing does not like unhappy customers.

By now the usual suspects in the form of team members were called again. This was the third time in a year that two team members were asked to drop everything they were doing and help on a rush job. By using the documented Elbonia and Monrovia experiences, the Dogbert team had a process defined, deliverables understood, and a good start on a questionnaire.

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

Subject Matter Expert: Mary Preston			Manager Responsible for Resolving Issue: Robert Notse		
Topic	Software Process Maturity (see questionnaire)				
Issue	Lack of software life cycle methodology				
Health	Red	**Yellow**	Green	Gold	
Risk	Showstopper	Critical Watch	**High Risk**	Latent Risk	Opportunity
Notes	The Dogbert Project team does not use a common repeatable, documented software development methodology. There is a company standard, but it has not been used by this team. As a result, key deliverables are missing, such as a current project plan, team roles and responsibilities, change requests, and outstanding issues. The assessment team recommends using the company standard immediately and preparing the key missing deliverables. Otherwise the project has a high risk of failure and will probably miss its key milestone dates.				
Estimated Completion Date	December 12, 1999				

Figure 1. *Example Dogbert review issue/problem rating template.*

Process Improvements

The Dogbert team did something very different from the previous teams — it changed the rules of how interviews were conducted. This proved to be the most important lesson learned in the review. When CMM assessors, auditors, and others conduct interviews, they usually do not identify the source of their findings (a person's name). The Dogbert team agreed to not only identify the subject matter expert for each issue, but also identify the manager responsible for resolving the issue. The review team and the people interviewed jointly crafted definitions of findings, issues, and due dates. They had to agree on the wording of the topic and corresponding issue, and its health and risk ratings. Technology was a big help with the use of a laptop and portable projector to show the findings on-screen. Figure 1 is an example of the template used to document issues, problems, and other findings, including positive ones.

Another process improvement the Dogbert team made was to define both health and risk ratings for the project. Health ratings are shown in Figure 2. Health ratings, in stoplight format, indicate whether there are major issues to be addressed and how well the activities are progressing per the schedule. Risk ratings have familiar titles such as showstopper, critical watch, high risk, latent risk, and opportunity. See Figure 3 for some examples. Risk ratings reflected impacts on the system and the customer if a problem was not resolved. Use of both ratings gave a clearer idea of the possible pitfalls for the IS project.

In addition to the commonly used red, yellow, and green ratings, the team added a gold rating. This process improvement proved to be invaluable since it recognized what was being done right. When the team presented its findings, the gold category was the first one discussed and it helped put the review in a positive frame of mind. After all, how often does a review recognize the positive things a project team is doing?

The Dogbert team successfully finished its review, received an award, and returned to work. But three of the team members realized the need to do something more than archive results from another review. The Dogbert team sponsor also recognized the need to "bottle" the team's experiences. The SRP Web site was about to be born.

The Boeing Intranet and the SRP

The Dogbert team created more than 100 electronic files containing everything from deliverable examples, presentations, and process descriptions, to intermediate deliverables, and more. In addition, there were the previous files from the Elbonia

Figure 2. *Dogbert review health rating criteria.*

Rating	Meaning	Criteria
RED	Not Acceptable	Unsatisfactory condition, or • There is an impact to the plan, commitments are not being met. • The end-item schedule and deliverable will not be met. • Required plans are not yet developed (no plan).
YELLOW	Partially Acceptable	Marginal condition, or • Concerns and/or a potential situation exists that may impact the plan deliverable. • Original commitment is in jeopardy and elements of the plan are not being met. • The end-item schedule and deliverable are at risk.
GREEN	Acceptable	Satisfactory condition, or • A customer-agreed-to plan consisting of a work statement, deliverables, and a schedule is in place with all activities authorized. • Commitments are being met and there are no anticipated problems.
GOLD	Noteworthy Status	Well done. Meets or exceeds expectations.

Category	Criteria
Showstopper	The system will • Not work. • Not be able to maintain an acceptable level of performance. • Create unacceptable downtime. • Create unacceptable data integrity errors.
High Risk	The system will • Require a high support effort. • Significantly degrade response and restoration time.
Opportunity	Opportunities for continuous quality improvement will lead to improvements and will not cause problems.

Figure 3. *Dogbert review risk rating criteria (selected ratings).*

and Monrovia reviews scattered about.

After the Dogbert review was done, three team members met to figure out how to capture all of this valuable material. Over the next few weeks several options were reviewed, but one emerged over the others — create a new Web site on the Boeing Intranet. All three team members had used the world-famous Boeing web and one of the members was responsible for several software engineering Web sites. They agreed to develop it in their spare time.

All the pieces came together, including a web developer who was between projects. The SRP site began in the fall of 1998 and within a couple of months the initial site was done. The site was not fancy and the material was basic, but it represented a collection of best practices that could be used by anyone doing reviews, assessments, or audits of projects, organizations, or suppliers. In fact, it could be used for just about any type of review even though the primary audience was software engineering practitioners.

The Production SRP Site

The site has evolved to one with more than 100 deliverable examples, lessons learned, and related links. The home page describes the general categories of IS reviews for which the site was built:

- assessment of suppliers of software products and services
- system production readiness
- project management of the IS project
- technical oversight and architecture for the system

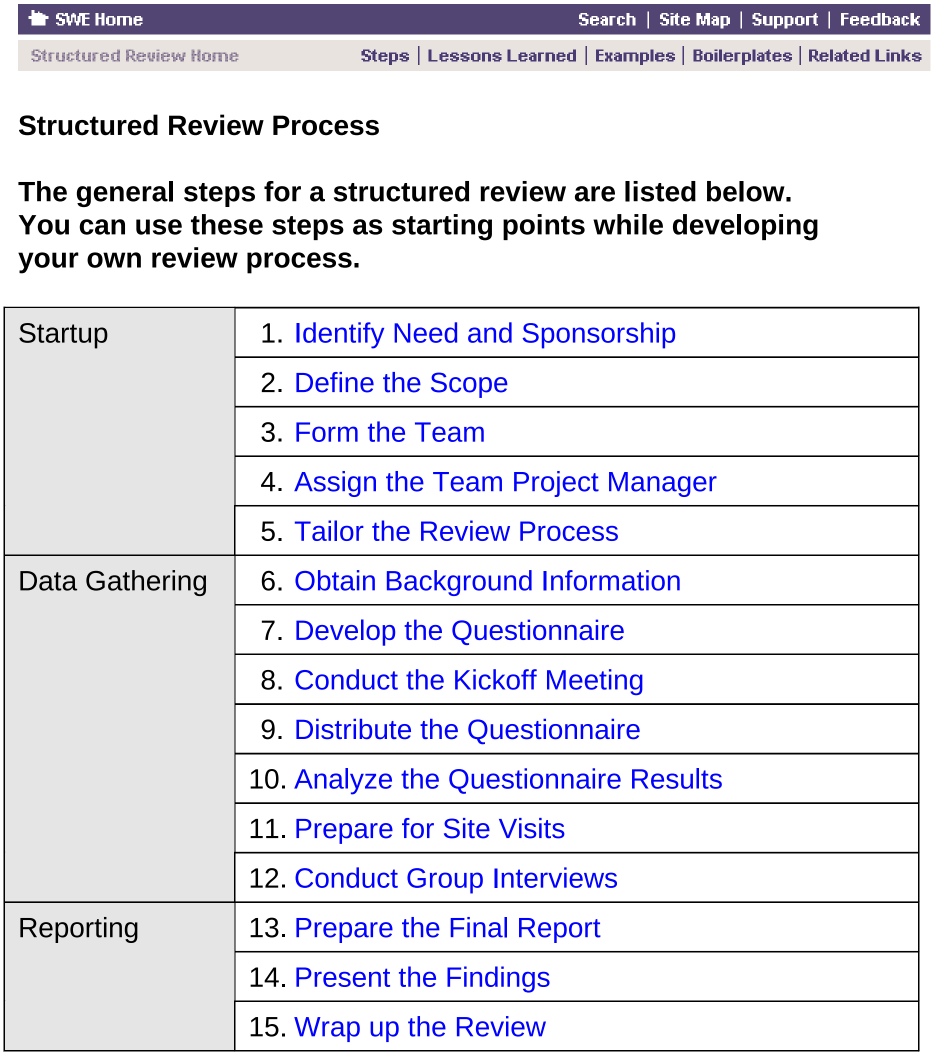
The review steps are basically the same whether you are reviewing an IS project or conducting other types of reviews. The navigation bar shown in Figure 4 lets the user go directly to any one of 15 steps in a review, look at deliverable examples for each step, select a boilerplate (template) for the deliverables, read the lessons learned for each step, or go to links to relevant Boeing and external Web sites.

Internal web links include Offshore Computing Support (they assess suppliers), production readiness reviews, computer disaster preparedness, sites for assessors, and other sites with tools or information of value to reviewers.

The Software Program Manager's Network is a popular external link, since it has free guidebooks invaluable for anyone conducting reviews: *Project Breathalyzer* (how to get a quick look at project health), *The Book of Software Management Questions* (good for program managers to understand what makes projects successful), and *The Little Book of Bad Excuses* (common excuses you hear from people being reviewed). There are more guides and all are free via the web.

One of the benefits of the site is that it allows a user to enter at any point in

Figure 4. *Key index page for SRP Web site.*


SWE Home
Search | Site Map | Support | Feedback
Structured Review Home
Steps | Lessons Learned | Examples | Boilerplates | Related Links

Structured Review Process

The general steps for a structured review are listed below. You can use these steps as starting points while developing your own review process.

Startup	1. Identify Need and Sponsorship
	2. Define the Scope
	3. Form the Team
	4. Assign the Team Project Manager
	5. Tailor the Review Process
Data Gathering	6. Obtain Background Information
	7. Develop the Questionnaire
	8. Conduct the Kickoff Meeting
	9. Distribute the Questionnaire
	10. Analyze the Questionnaire Results
	11. Prepare for Site Visits
	12. Conduct Group Interviews
Reporting	13. Prepare the Final Report
	14. Present the Findings
	15. Wrap up the Review

the review process. For example, a team may have been formed, but it needs help developing a questionnaire. Team members can go directly to Step 7, Develop the Questionnaire, to see how others have done this step. They could reuse many parts of the questionnaire template or they can view lessons learned from other reviews.

If a review team has just been formed, it can tailor the 15-step process to meet its needs. For example, it may not do a questionnaire and will only conduct interviews. The team can eliminate the unneeded steps to come up with its own process. A CMM assessor could use the site to complement the guidance from SEI since the questionnaire exists. The "how to" steps are included in the SRP site and help any reviewer, experienced or new.

Early Results

Since the site went into production it has been used in several successful reviews. In one case — another foreign supplier assessment — the review flow time was cut in half due to the use of reusable processes and deliverables. The team met a tight schedule even though many of the team members were called away on other special assignments.

In another case — review of an IS organization — the site was used to develop the review process and identify deliverables. Again, hundreds of hours were saved due to reuse. In a third case, the SRP site was used to develop a process to conduct reviews of potential suppliers of computing services. Savings in time, cost, and schedule were considerable without sacrificing quality.

Next Steps

Many additions are being made to the Web site, including more examples, boilerplates, lessons learned, and links. A new category, Tools, will be added since there are a number of commercial and in-house tools that help reviewers. The site has had rave reviews throughout the company and communications about SRP will continue in the form of presentations, news articles, and other means. Several Boeing divisions are using the site.

Conclusion

Without the SRP site, many review teams would have continued reinventing processes for IS reviews. But now there is an alternative to starting from scratch. The SRP site has captured years of experience in the form of reusable, tailorable processes, deliverables, and tools. Creating the SRP site took minimal effort and was completed in two months. The benefits are still coming in, far exceeding the initial investment. The site will continue to be improved as experiences from future reviews are added. ◆

About the Authors

Darrell Corbin is an associate technical fellow in Companywide Software Engineering at the Boeing Co. He has more than 31 years experience in business and engineering information systems. He provides Boeing-wide support in software methodologies and software process improvement. He is a project manager for the SRP Web site. He has bachelor's and master's degrees in economics from Washington State University.

The Boeing Co.
P. O. Box 3707, #MC 7L-16
Seattle, Wash. 98124-2207
Voice: 425-865-5939
Fax: 425-865-2663
E-mail: darrell.s.corbin@boeing.com

Russ Hamerly is project manager in Distributed Computing Program Integration at the Boeing Co. He has more than 25 years experience in electronic switching, networking, and computing systems. He provides Boeing-wide support in program management, system management, and emerging workstation technologies. He is a subject matter expert for the SRP Web site. He has a bachelor's degree from the University of Washington and a master's degree in business administration from the University of Notre Dame.

Roger Cox is a senior manager in engineering information systems at Boeing Commercial Airplanes. He has more than 32 years of information technology experience, including the last 20 years at Boeing. Cox also serves as an adjunct professor of software engineering at the University of Washington and an adjunct professor of information systems in the graduate program in the school of business at Seattle Pacific University. Cox holds bachelor's degrees in engineering physics, mathematics, and computer science, and master's degrees in both physics and computer science, is doing doctoral studies in computer science, and holds an advanced management certificate.

Dr. Kenneth Knight is professor of information systems management at Seattle Pacific University. Knight has been active as an author, professor, manager, and consultant in the information systems area for 35 years. His previous academic positions were in the School of Business at The University of Texas at Austin and Stanford University.

A Multi-Site Software Process Framework

Ralph E. Porter Jr. and Deborah A. DeToma
GTE Government Systems Corp.

Today, software process leads to the new paradigm, "Better, faster, cheaper — through continuous software process improvement (SPI)." However, developing a standard software process might be considerably easier than the task of rolling it out to multiple locations across the U.S. and perhaps overseas, a concept we refer to as "zero geography." GTE Government Systems Corp. (GSC) successfully implemented its SPI program across North America using a framework based on the 3 C's: commitment, continuity, and communications. With zero geography, GSC was able to leverage its existing assets, accelerate schedules, and minimize investments while reaping the full benefits of SPI.

The Challenges to SPI

The challenges to the software community come in many forms. However, one of the most promising trends in software development presents itself in the form of SPI programs. In the past, the paradigm was, "Better, faster, cheaper; pick any two." However, adherence to a standard software process has brought about consistency, improved productivity, and reduced error rates. Software process, in fact, leads directly to the new paradigm, "Better, faster, cheaper — through continuous software process improvement."

GSC has successfully implemented SPI across a wide array of business and customer types and locations. This paper will identify the SPI challenges we have faced, the impacts the Department of Defense (DoD) acquisition reform has played, the 3 C's framework that we have and continue to use to implement our corporation SPI plan, the benefits reaped, and some lessons learned. This framework was essential to ensure that each geographic site maximized reuse of existing SPI assets, that opportunities for collaboration are identified to minimize effort, and that invaluable lessons learned are shared. In this article, when we use the term corporation, we are referring to GSC and when we use the term parent company, we are referring to GTE.

Diversity — Business, Technology, Customers, and Geography

As with many high technology companies, GSC is committed to SPI. Similar to other high technology companies, GSC has many software development sites, geographically spread across numerous business units. However, unlike other such companies, GSC does not concentrate only on its largest sites; all GSC sites with any significant development are under the GSC SPI program. GSC's diversity, however, extends far beyond just geography; there are significant differences in business sets, customer communities, domains, technologies, tools, and methodologies.

GSC consists of four major divisions and a headquarters organization. Three divisions and the headquarters organization involve software development — Communications Systems Division (CSD), Electronic Systems Division (ESD), Information Systems Division (ISD), and the GSC Information Technology organization. Each of the three divisions, with multiple sites, are involved in all aspects of development, from new developments to modifications, ports, enhancements, and maintenance. This means that the corporation's standard software process must address a very broad range of programs and tool environments. In other words, the process must be tailorable. A full description of the corporation's diversities of business, technology, customers, and geography are presented in [1].

The trend towards use of a multi-site software framework has become more commonplace in the past few years. This has been driven largely by the consolidations in the aerospace industry. For example, Lockheed Martin today is comprised of mergers and acquisitions of Lockheed, Martin Marietta, Loral, GE Aerospace, GD Space and Fighters, IBM Federal Systems, and Unisys — all since 1992. A corporate SPI program in such a large, geographically dispersed company requires an infrastructure to support it.

The Solution Set: The 3 C's: Commitment, Continuity, and Communications

To address the broad range of diversities and challenges, the corporation has employed the 3 C's as the basis of our SPI program.

Across the parent company, this had been on a division-by-division basis for many years, starting in the late 1980s. However, these divisional commitments tended to be narrowly focused on those organizations where software products were delivered externally. Their customers frequently determined the requirements of the Software Engineering Institute's (SEI) Capability Maturity Model® (CMM) that would apply to their projects. Due to the information technology industry's poor past performance and project failures, GTE began to focus on improving its ability to deliver quality software on time and within budget. This led to a 1996 study on software quality and focused on internal and external software systems. GSC, through its involvement in the DoD market, had its SPI program well under way when Kent Foster, president of GTE, levied process improvement goals on the entire parent company following this study. This corporate SPI goal is strong evidence that commitment can start at the top level of management. Foster presented well-defined goals for the corporation associat-

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

ed with software process improvement and software acquisition. These goals have hard dates associated with them — to achieve Level 3 by December 31, 1999.

One — Commitment

To facilitate parent company-wide commitment, Foster formed the Software Quality Initiative (SQI). A small, but highly CMM-knowledgeable team was put in place to establish goals and guidelines as well as track and report status. Below that, key individuals were identified to coordinate the SPI activities across large business units (e.g., the National Operations, Wireless, and GSC). The president of GSC rapidly embraced and supported the initiative as the corporation already had a software quality-focused team in place.

Below the SQI team, the lead "SPI zealots" of each major organization (e.g., the GSC Electronic Systems Division) were assigned to a software process leaders group, which meets quarterly. The maturity, experiences, and assets of more mature organizations are used to leverage less mature organizations more rapidly than if each site were on its own and developing its SPI program from scratch.

Each organization established SPI objectives that are used to support the GSC business areas. The intent was to meet request for proposal needs and requirements and provide our corporation with a competitive edge in acquisitions. In the area of program performance, we set out to meet or exceed customer needs and requirements, provide our customers with better, faster, and cheaper products and improve the quality of our products and services with an overall goal of zero defects.

This level of commitment has ensured that each parent company organization has a clear objective in terms of the maturity of its software process. The SEI Level 3 requirements must be demonstrated through an approved assessment method (e.g., CBA IPI).

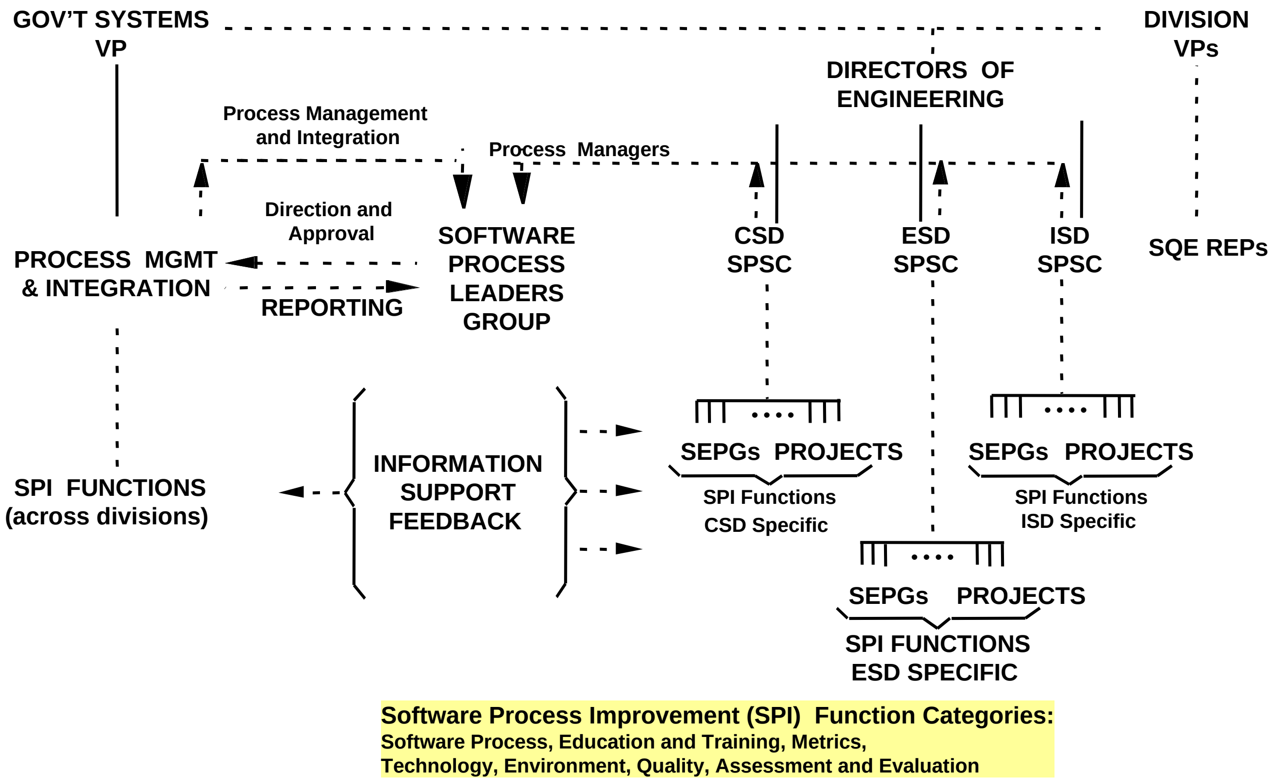
Two — Continuity

The second aspect of GSC's SPI program is continuity, in terms of continuity of the processes used on our software development programs. The GSC SPI focus includes the process management and integration (PM&I) organization that reports to the corporate level. The PM&I's role is to ensure that communications, leveraging, and status reporting routinely occurs. The PM&I organization is ultimately responsible for the satisfaction of the parent company corporate goal. PM&I works closely with the division SPI organizations, whose role it is to work with the projects within the division.

Although the above infrastructure may appear large and cumbersome, it is very effective and does not require hoards of people to implement. Within our corporation, there is only one individual who is the PM&I representative. This person coordinates the corporate process plans, which include SEI and ISO 9001. At the division level, a single representative is appointed to be the point-of-contact with his/her division and across the corporation.

Each division has a software process steering committee, a software engineering process group and a series of process action teams (PATs). The PATs are the mechanism that we use to make SPI real to the engineers. There is great employee participation

Figure 1. *Corporate software processes.*


GOV'T SYSTEMS VP
DIVISION VPs
DIRECTORS OF ENGINEERING
Process Management and Integration
Process Managers
Direction and Approval
PROCESS MGMT & INTEGRATION
REPORTING
SOFTWARE PROCESS LEADERS GROUP
CSD SPSC
ESD SPSC
ISD SPSC
SQE REPs
SPI FUNCTIONS (across divisions)
INFORMATION SUPPORT FEEDBACK
SEPGs PROJECTS
SPI Functions CSD Specific
SEPGs PROJECTS
SPI Functions ISD Specific
SEPGs PROJECTS
SPI FUNCTIONS ESD SPECIFIC
Software Process Improvement (SPI) Function Categories:
Software Process, Education and Training, Metrics,
Technology, Environment, Quality, Assessment and Evaluation

and interest in the PATs. A detailed description of the common SPI organizations within the corporation, and the roles of all the participants, was presented in [2].

Tailoring is the key to making our standard process work for projects of all shapes and sizes. We have four models that are used for our projects. With relative ease, we can select an appropriate model for use on the project. In [3], the four software process models (full, intermediate, basic, and special) are described; specific tailoring guidelines were presented.

Further, a tailorable CMM-based software process mini-assessment method [4] to meet the demands of our parent company SQI. This corporate mini-assessment method has evolved into a well-defined process, with components that are tailorable to meet the specific objectives and needs of our organizations. The corporate mini-assessment method includes guidelines for planning, preparing for, and conducting a mini-assessment in three to four days. Options for tailoring the approach and reducing risk in order to ensure accuracy and completeness are provided. Reusable mini-assessment assets are utilized to improve efficiency and ensure consistent application across our corporation. The mini-assessments conducted to date have been very well received by participating organizations due to the method's flexibility, accurate results, and ability to accelerate the momentum for process improvement.

Three — Communications

Just as important as setting goals is the communication of those goals to the organization. Otherwise, the "grass roots" efforts required to achieve those goals never get started and the initiative dies a slow, miserable death. One of the areas that we needed to overcome was the distance challenge. We found that there were enablers to address this tough issue. The corporate SPI focus includes the PM&I organization that reports in at the corporate level. Its role is to ensure communications, leveraging, and status reporting occurs. The PM&I is ultimately responsible for the satisfaction of the parent company goal.

In 1989, GTE formed a Corporate Assessment Team, which is comprised of authorized lead assessors and CMM-trained team members from throughout the corporation. This team is managed through the PM&I organization. There are quarterly meetings of the Software Process Leaders from throughout the corporation. The purpose of the meeting is to review status and provide a mechanism for sharing. There are other workshops, such as Metrics and Tools, that have helped in defining and refining the standards in these areas. Process tools have been deployed, such as FastAssess, LBMS, and CMMLive.

Web-based communications has increased, including web front-ends to division process asset libraries. E-mail, phone, teleconference, and videoconference are used to supplement our face-to-face meetings. The *GTE News-GS Edition* weekly newsletter has been used to spread awareness of SPI by featuring news articles on the recent happenings and accomplishments in the divisions. Presidents, vice-presidents, and directors, as well as process personnel have provided articles. Also, articles for division SPI newsletters have been provided by program managers, software project managers, and practitioners from software and systems engineering, CM, QA, etc.

Celebrations and parties are also a part of this initiative, including picnics, barbecues, and a CMM fair bolstering a rousing game of "Stump the Process Expert."

Further, ESD, with headquarters in Mountain View, Calif., has invested heavily in the development of a collaborative work environment called InfoWorkSpace™, or IWS. IWS has been deployed to more than 1,500 Department of Defense customers (with a contract in place to deploy 13,000 more over the next three years) and recently was named "Best New Product 1999" in the workgroup/departmental software category at FOSE '99 Conference and *Government Computer News* [5, 6]. ESD's SPI plan includes the future migration to IWS of its software project management training and phase-specific development training. This will facilitate distance learning for any employee, anywhere in the world, using any desktop platform with a browser such as Netscape or Internet Explorer.

Benefits of the Framework

As a direct result of our SPI program, we have seen significant improvements in our productivity, quality, and predictability due to our emphasis on process. There have been cost reductions based on streamlining of processes. We have also experienced some qualitative differences including employee retention and hiring. It is more attractive to work in a more mature organization which has employee involvement in improvement efforts. The SPI work is becoming "real" to the people.

The success of our 3 C's framework is evident through our successful CBA IPI assessments in recent years; many of the sites have already achieved SEI Level 3 using this 3 C's framework to achieve zero geography:

- ESD-Mountain View, Calif.: Level 3, 3/94 and 9/97
- ESD-Tempe, Ariz.: Level 3, 9/97
- ESD-Thousand Oaks, Calif.: Level 3, 3/99
- CSD-Needham, Mass.: Level 3, 4/99
- CSD-Taunton, Mass.: Level 3, 4/99
- ISD-Chantilly, Va.: Planned 3Q99

Also, our corporation participated in the 1994 SEI return on investment (ROI) study [7] by providing actual program data. For the five-year period of the study, the results showed that productivity increased 37 percent in terms of source lines of code/hour, error reductions netted 55 percent less defects/thousand source lines of code, and the overall SPI ROI was 6.8. An internal division ROI study conducted in 1995 found similar results with their ROI being 7.8.

Other cost reductions have been seen throughout the corporation. The average software defect rate during system integration and test has been significantly reduced over time. Within one division, the level of formal quality assurance support has dropped from being 2.2 percent of the organization (based on head count) to under 1.8 percent (almost a 20 percent reduction). In 1997 that division tailored its software quality assurance (SQA) activities, taking advantage of the maturity of its peer review process, thereby reducing its SQA costs by 50 percent on its programs. In all cases, the improvements in our software process have increased quality while reducing costs, thereby reducing time to market.

Lessons Learned

Top management commitment is essential. Process zealots are required. It is very important to have people who really believe that the effort will make a difference. It behooves you to have people who are respected in the organization in these positions. This could mean the difference between a successful program and the perception of just another "quality initiative." We have experienced enhanced communication between sites and are promoting reuse of knowledge, process, tools, and people. Delivering products better, faster, and cheaper is now achievable using our zero geography approach to software process improvement. Finally, our multi-site software process framework has become "Our way of doing day-to-day business!" ◆

About the Authors

Ralph E. Porter Jr. is currently employed with General Dynamics Electronic Systems (purchased from GTE Government Systems Corporation on 9/1/99) as Director of Process & Quality Assurance for Electronic Systems (ES), with 10 sites directly involved in SPI activities. Ralph was managing the Software Process Organization at the time ES received their SEI Level 3 rating in March of 1994, September 1997, and March 1999. Currently he manages all activities associated with software process, software tools, ISO 9000, quality assurance, Y2K, and configuration management.

Ralph received both a B.S. degree in Mathematics (1972) and a M.S. in Statistics and Probability (1974) from Oklahoma State University. He is a member and Vice-President of the Silicon Valley Software Process Improvement Network (SV-SPIN) as well as Co-Chair for the SV-SPIN Metrics SIG. He has given numerous presentations on software process, metrics, and estimating to the Bay Area RoundTable, SV-SPIN, and SEPG conferences (1996, 1997, and 1999).

General Dynamics Electronic Systems
100 Ferguson Drive
Mountain View, Calif. 94039
Voice: 650-966-2023
Fax: 650-966-4024
E-mail: Ralph.Porter@gd-es.com

Deborah A. DeToma is currently employed by General Dynamics Communications Systems (purchased from GTE Government Systems on 9/1/99) as the lead in the Process Management and Integration organization. She has been directly involved in software process improvement activities at GSC for 10 years, specializing in Software Estimation, Cost Modeling, and Measurement.

Debbie received a B.S. degree in Mathematics/Computer Science from the University of Lowell and a M.S. degree in Computer Science from Boston University. She chairs the corporate-wide Software Process Leaders group, the GTE Software Metrics Workshop, the ISO 9000 Management Representative council, and has participated in the DoD Practical Software Measurement program. She is a member of the Boston SPIN.

General Dynamics Communications Systems
77 A Street
Needham, Mass. 02194
Voice: 781-455-3286
Fax: 781-455-5734
E-mail: Deborah.DeToma@gd-cs.com

References

1. Porter, R. and D. DeToma, "Software Process Improvement with Zero Geography!," SEI 1999 SEPG Conference, Atlanta, Ga., March 1999.
2. Dorsey, T. and D. McDonald, "Structured for Success: A Software Engineering Process Group Model," SEI 1996 SEPG Conference, Atlantic City, N.J., May 1996.
3. Porter, R., "Software Process Models — Tailored to Fit; Tailored for Success!," SEI 1996 SEPG Conference, Atlantic City, N.J., May 1996.
4. Pilch, C. and D. McDonald, "A Tailorable Software Process Mini-Assessment Method," SEI 1999 SEPG Conference, Atlanta, Ga., March 1999.
5. http://biz.yahoo.com/bw/990324/va_gte_1.html
6. http://infoworkspace.gsc.gte.com
7. Herbsleb, J., A. Carleton, J. Rozum, J. Siegel, and D. Zubrow, "Benefits of CMM®-Based Software Process Improvement: Initial Results," CMU/SEI-94-TR-013, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pa., August 1994.

A Problem-Based Approach to Software Process Improvement: A Case Study

Johanna Rothman
Rothman Consulting Group, Inc.

Organizations struggle [1] with their process improvement efforts for a variety of reasons. Perhaps the most common struggle pattern is to take a long time developing a general understanding of their processes and then trying to define all possible alternatives in the product development process. This pattern leads to large, unmanageable, unreadable, and incomplete [2] process documentation.

This paper is a case study of one organization that minimized the struggle by taking a different approach to the development of their product development process.

Introduction

The organization described in this paper was a 12-year-old company, formed out of two startups. It created and sold graphics products. We will call this organization "ExtendIt." ExtendIt employed about 150 people worldwide. The product development staff was split into two locations: about 50 people in the Boston area office, and about 20 people in a European office.

ExtendIt was in a typical chaotic state — most of senior management did not understand the software product development process. Engineering management did not know where or how to begin, project management and product management were nonexistent, and engineering processes were completely inadequate for product development and testing. Projects were planned for four to eight months, but typically took 13-18 months. Even at the end of the extended development time, ship decisions were generally based on emotional reasons to ship, not objective reasons. For example, management made the decision to ship a major release because the developers were too tired to continue the 80-hour weeks, not because the project met the ship criteria. In fact, that particular project did not have all the expected features, so the developers continued to work long hours to get the features into the follow-on release.

Approach to Process Improvement

A new CEO started at ExtendIt and changed the product strategic vision and sales model. Based on the new goals, it was clear that the organization had to change how it developed products. It was not possible for this geographically dispersed engineering organization to meet the new goals without changing their practices.

Senior management had already agreed to decouple releases from project development, which is a typical concurrent engineering approach to product development. This would be known as the "release train," a quarterly plan to ship products[1]. Projects at a certain point in their development would be eligible to be loaded on the train and be shipped. Projects would not be shipped unless ready. To meet the release train goals, ExtendIt formed small independent projects.

A software engineering process group (SEPG) [3] was formed in May 1997, with the original plan that the process definition and design could be completed by the end of July 1997, a total of eight weeks. The SEPG consisted of engineering management such as the vice president (VP) of engineering, the documentation manager, development, and release engineering managers; the director of program management, and an outside consultant — a total of seven people. The initial roles of the people on the SEPG were:

- The VP of engineering was the facilitator between the SEPG and organization at large.
- The documentation manager served as the chairman of the SEPG and provided expertise about documentation processes.
- The two development managers and the release engineering manager provided expertise about current processes and how they could be changed.
- The program management director provided specific engineering expertise and general organizational expertise about product development.
- The consultant provided planning and facilitation for the SEPG meetings in addition to process and product development expertise during the process design.

Like many organizations, the SEPG planned to roll out the process definition and templates to the organization à la the hole-in-the-floor model of change [5][2]. The rollout milestone was planned for August 1997. After the initial SEPG effort, engineering management was to carry out ongoing process change.

This SEPG forgot one thing — change is not successfully rolled out to organizations [5]. People have to integrate the changes into their daily lives for the change to be successful. Although this SEPG did not anticipate this, changes were introduced and integrated into the organization in a most fortunate and successful way.

Problem Statement

The SEPG began by discussing what had to change. Using brainstorming, they identified 29 issues. Then they used affinity grouping to sort the 29 issues into nine "buckets"[3]. Each SEPG member cast three votes, and voted on their top three issues. They took the top 80 percent of the problems and threw away the lower 20 percent. The result of this analysis were the following six problem statements:

1. The product development process was not documented. The process was not uniform among projects.
2. The functional specs/design specs were not separated. Because the functional description and the design was intertwined, some parts of the system were not well-defined and the test planning effort was insufficient.
3. Vague marketing requirement documents (MRDs) told development how, not what, to do.
4. Development's intake of market requirements were not well defined or controlled. This was really an organizational problem — getting a single point of contact for discussing issues.
5. Too many off-process interruptions. The engineering staff was interrupted or dragged off to work on other issues. There were no organization-wide rules about how to get consulting from others.
6. Managing to a schedule was a problem. People did not know how to manage their own time, or how to rank their activities.

Each SEPG member wrote six descriptive sentences describing each problem as it appeared to or affected each person. The SEPG called this their "6x6" matrix, for six sentences about each of six problems. Everyone's sentences were gathered into a concept matrix, with each major item on the left, and the relevant issues on the right. The SEPG then grouped the problems into subcategories, to organize the issues. (See Table 1 for a representative portion of the final set of problem statements.)

The final concept matrix has a generic problem statement, specific issue, and examples of how each issue affected the organization. The SEPG then made a critical decision — the SEPG decided to focus its work on just the six problem statements above: documenting the product development process; separation of functional and design specs; specific MRDs; how development took in requirements; managing interruptions; and managing to a schedule. This focus provided these main benefits:

- SEPG modeled problem-solving behavior — Not every decision was correct in hindsight, but the problems were discussed in context of the problems the SEPG was trying to solve. The decisions and the decision-making process were accessible to the organization.
- SEPG practiced problem-solving skills — The managers were on the SEPG. They had a chance to practice their problem-solving skills in an environment of their peers, before trying them out on a project. This included practice using the traditional problem-solving skills and tools, such as brainstorming, affinity grouping, and facilitating discussions of diverse ideas.
- Focused SEPG work — ExtendIt was working towards a rational way of doing business, not towards public certification or assessment. Using the business as incentive for the process improvement activities was understandable by the management and technical staff.

Table 1. *Portion of concept matrix describing problem statements.*

Product Dev. Process not documented	A: Central Reference Required	1. There is no "playbook" which can be given to all employees, so they know the process for developing software. 2. Missing the "what, when and who" for product development maintains our current (perhaps our past) operating procedures (i.e., controlled chaos) 3. Without a documented product development process, there is no real way to determine where we are in the development process (there is no "starts with" or "ends with" statement or entry or exit criteria). 4. …
	B: Common Terminology	5. Terminology is imprecise, e.g. "Alpha" means something, but not the same thing to different people. 6. …
	C: Phase Definition/Criteria	7. There is no current opportunity to define project success criteria. 8. There is no current opportunity to know when a project is complete. 9. There are no clear phases, with entry/exit criteria to know what is done and what is not. 10. …

Intermediate Results

The VP of engineering and some SEPG members felt very strongly that some aspects of product development could not be planned. The VP wanted the SEPG to take an approach to process definition that facilitated reasonable things for reasonable people to do. The SEPG would then incorporate management reviews into the process that were sufficient to inform management, and enable management to take appropriate steps. In addition, the process documentation would give general problem-solving guidance. (Online documents describing useful meeting techniques and project management techniques were part of the final deliverables.)

The SEPG approached the process definition work as if it were an engineering project. The work started with a strawman five-phase process:

- Concept/Requirements
- Design/Definition
- Coding/Implementation
- Validation/Verification
- Manufacturing/Ship

Starting from its charter, the SEPG initially refined its concept (Concept/Requirements phase). The SEPG took the time to define its requirements and an initial project plan, to clarify project completion and success criteria. To clarify and define SEPG deliverables, the initial SEPG project plan used the five phases above.

During the design and definition phase, the SEPG defined the functional specification and design. The SEPG made an initial cut at the phases, figured out what the necessary documents had to be, and where the review points were.

The implementation phase consisted of the detailed design of the process description, and generating the flow charts and words to describe it. To get early testing, and to get engineering buy-in, the SEPG held focus groups to discuss each phase. Getting early engineering input had these benefits:

1. The SEPG's work was visible to the organization. In fact, parts of the organization were able to test the process by using pieces of it on ongoing projects. Doing this early testing has some ramifications:
   - The SEPG could see if the people who were supposed to use the process would actually use it.
   - A number of issues arose during these focus groups. The discussion around these issues allowed the SEPG to change and simplify the process.
2. The SEPG was able to gain substantial experience in presenting the process to the organization. When the focus group was confused, the SEPG could test how the focus group understood different descriptions.
3. The SEPG walked the talk of "early and often" review and testing. By having their work held up for review and verification, it was easier for the engineering staff to buy into frequent reviews and early testing.
4. Using an evolutionary process design meant the SEPG did not have to get everything right the first time. The engineering organization could see this, and see the relevance to their work.

At the end of the implementation phase, the five-phase product development process had evolved into:

- Concept/Requirements
- Design/Definition
- Coding/Implementation
- Validation/Verification
- Product Qualification

Disadvantages of this Approach

The SEPG worked very quickly, so it was hard for some people to integrate the changes to how they thought. Although the SEPG members did not have trouble with the concept of iteration, some had trouble with their ability to iterate their thoughts quickly. These SEPG members were thrown into chaos [5] with almost every meeting, and had a difficult time adjusting to the pace of change. Change can be painful to the people involved.

During the SEPG's work every member had to closely examine and change or give up closely held ideas about product development. Changing your mind about something when you do not have direct experience with its potential for success can be very hard. Some of the SEPG members were quite reluctant to change how they worked, even when they admitted their current patterns were not working.

For example, the SEPG intellectually understood that inserting a milestone at the beginning of the Coding/ Implementation phase to verify the release criteria against each project's criteria made sense to everyone. Some SEPG members were concerned that these release criteria would be fixed too soon and would be nonnegotiable. They were concerned that they would be forced to develop the wrong product. The rest of the SEPG, from experience, realized that clarifying release criteria before the code is finished is one easy way to make sure that the product under development is the correct product. The reluctant SEPG members were concerned because they had no experience with the success of release criteria. They knew their current methods were inadequate, but were reluctant to agree to something they had no direct knowledge of. As an SEPG, we agreed to conduct mini-retrospectives during the first few projects, to check on this and other points in the process.

Some of the SEPG members also had trouble changing their meeting behavior. Some team members were stuck in legacy behavior, using the same assumptions that had created the problems. One assumption was that all decisions were open to more discussion and change after the decision was made. It was impossible to make progress when all decisions could be revisited at any time by anyone. Consequently, the SEPG remained stuck in the "storming phase" of team development [4]. After discussing these problems with the SEPG chairman, the consultant requested the VP of engineering attend some team meetings. The presence of the VP acted as an inhibitor to "business as usual," and allowed the team to make appropriate decisions and move forward. In the case of the SEPG's decision-making, the VP verbalized the SEPG's responsibilities and the time to deliver on those responsibilities.

Results of Using the Process

SEPG Results

The original dates were very aggressive (an eight-week schedule), and were not met. Missing the original dates created these results:

- The SEPG was able to practice iteratively replanning its schedule. This experience was directly applicable to normal engineering projects.
- After the first milestone was missed, the SEPG practiced testing its work focus. Were members working on the most time critical and valuable item?
- The SEPG clarified its tradeoff decisions and decision-making process. It created a "Pending Bin" to place ideas and issues that were relevant to address, but not now.

All these issues emulated typical challenges of a product engineering project. The SEPG gained the understanding that its work was a process development process. The end result was not a saleable product, but it was a process where similar tools and ideas were useful.

Product Development Results

Initially, the engineering staff was concerned about changes to how it was expected to do product development. At the initial overview presentation of the release train, the engineering staff was confused by terminology and how to do what, because the specific changes to the process were not rolled out. The SEPG started its work after this initial presentation.

To get buy-in from the engineering staff, the SEPG started focus groups to discuss the process steps and then the templates in group meetings. The SEPG chose one SEPG member to present each life cycle phase to the focus group. The focus group would ask questions, and the designated SEPG member answered the

Typical problem	Avoidance
Process improvement effort takes a long time.	Focus the SEPG's efforts on the problems that need to be solved now. The engineering staff was not only willing to use the process; it demanded it on projects.
Define all possible steps in the process.	The process provides reasonable guidance and specific criteria for escalation to management. Project participants use the process as a guide. They use their judgement about how to deal with problems, until the problems need to be escalated to management.
"Big Honking Binder" syndrome: the size of the documentation overtakes the process definition.	One-page process flows with one-page descriptions. One- and two-page document templates are part of the process definition. The whole process document is 20 pages.
Technical staff is suspicious of process development process and reluctant to adopt outcome.	Test the process with staff as it is developed.

Table 2. *Lessons learned by ExtendIt.*

questions. The rest of the SEPG staff took notes about the presentation and the questions. When there were many questions, the SEPG generally redesigned the process to make it easier to understand, easier to implement, and more streamlined.

After the process was reviewed in the focus groups, the templates (plans and specification documents) were reviewed in focus groups. The SEPG used the same process: one SEPG member presented the material, and the focus group commented on the material.

By the end of the focus group activity, all the senior staff in engineering had seen parts of the process and the templates. Because the engineering staff helped create and review the process and the templates, the senior staff led the rest of the technical staff to adopt the process. At the next general presentation, the overall process was discussed. The engineering staff understood the process and the templates and it had been made clear what they had to do and when.

Lessons Learned

ExtendIt employees learned a tremendous amount from these steps to process improvement: a process improvement process. They were able to avoid some typical process improvement problems shown in Table 2.

Conclusions

This process improvement process was very effective. It consisted of first determining the problems that needed solving, then developing a process that illustrated the way to do the general case, and a set of problem-solving skills. About eight weeks after the SEPG formation, the SEPG members began to work differently. The SEPG thought about their deliverables to each other in a more complete way, i.e. how people could use what they developed, and the effects of their deliverables on other deliverables.

The biggest organizational change was that the managers and technical staff thought differently about how to do their work. They started to plan for the reasonable case, and created a risk assessment and management plan. This had the desired effects of creating simpler project plans, and pushing risk assessment into the organization.

A small process description seems to be adequate for the present for this organization. The process description contains five pages of flowcharts, about four pages of definitions, and about five pages describing the process and general problem-solving techniques. In addition, there are templates for each document the engineering staff produces.

ExtendIt has been using this process for almost a year. It has successfully produced three quarterly release trains. The technical and management staff has tested the process, and for now, it works.

ExtendIt has had a difficult time escaping from its startup phase. The new CEO and senior management are determined to make the company a success. From a product development perspective, the organization can now deliver products on time and within budget, with the requested features. Using the release train to chunk the features into smaller independent projects, and by creating the expectation that the organization would deliver multiple products over the course of the year, ExtendIt is operationally poised to succeed. ◆

About the Author

Johanna Rothman speaks, writes, and consults on managing high technology product development. She works with her clients to increase the effectiveness of their managers, helping them ship the right product at the right time, and hire and retain the best people. She has more than 20 years experience in software engineering and management and is part of the clinical faculty of The Gordon Institute at Tufts University. Rothman holds two American Society for Quality certifications: Certified Quality Auditor and Certified Software Quality Engineer.

38 Bonad Rd.
Arlington, Mass. 02476
Voice: 781-641-4046
Fax: 781-641-2764
E-mail: jr@jrothman.com
Internet: http://www.jrothman.com

Acknowledgements

In addition to the anonymous reviewers, I thank the following reviewers for their help and substantive comments: Don Gray, Brian Lawrence, Sue McGrath, and Jerry Weinberg.

References

1. Hayes, Will and Dave Zubrow, *Moving on Up: Data and Experience Doing CMM-Based Software Process Improvement*, CMU/SEI-95-TR-008, 1995.
2. Murphy's Law. Specifically, "Whatever can go wrong will, at the worst possible time."
3. Humphrey, Watts, *Managing the Software Process*, Addison-Wesley, Reading, Mass., 1989.
4. Scholtes, Peter R. Joiner and Streibel,

The Team Handbook, Joiner Associates, Madison, Wis., 1996.

5. Weinberg, Gerald, *Quality Software Management: Volume 4: Anticipating Change*, Dorset House Publishing, New York, 1997.

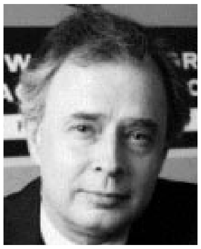
Notes

1. Companies who have the need for parallel development of multiple releases use this concept. Although Sun has implemented this differently, the release train idea described in http://solaris.license.virginia.edu/sun_microsystems/workshop4.2_docs/teamware/solutions_guide/casestudy.doc.html No. 8868 is similar in concept.
2. The hole-in-the-floor model of change: Some set of people upstairs develops the perfect system. The change plan consists of drilling a hole in the floor. The system is dropped through to the people below. Supposedly people instantly change to the new system. Unfortunately, people generally cannot change without integration and practice.
3. Affinity grouping is the activity of creating sets of similar ideas together under one theme. In this case, we wrote each problem on a sticky note, silently organized the sticky notes into groups, and then named each group.

"High Leverage Best Practices..." continued from page 14.

Management. This identifies and describes in detail the 16 Critical Software Best Practices. SPMN, in coordination with the Airlie Council, also developed a related implementation handbook and compact disc, that is in beta testing, and that is being enriched through a review process by a tri-service group of program managers. These materials, along with related briefings, video clips, and other material are available without charge from the Software Program Managers Network at www.spmn.com. ◆

About the Author



Norm Brown is the change agent for more than 200 Army, Navy, Air Force, and other defense programs. He founded the SPMN, which conducts benchmarking of commercial best management and technical practices, and provides technical and management consulting support to these programs, and which has some 10,000 members. He founded the Airlie Software Council, led the Defense Department's Software Acquisition Best Practices Initiative, and was a member of the DoD Software Management Review Board. He chaired the Joint Logistics Commanders Group on Technical Data and Computer Software and the group that drafted the Federal Acquisition regulations on Technical Data and Computer Software, and numerous other groups and committees. He was responsible for the review of more than 300 defense software acquisition projects. He served in numerous acquisition selection boards and advisory positions to the Secretary of the Navy, Deputy Secretary of Defense, and other defense officials. Brown was the software manager for a number of real-time Navy weapons systems and was a commercial software developer.

Software Program Managers Network
P.O. Box 2523
Arlington, Va. 22202
Voice: 703-521-5231
Fax: 703-521-2603
E-mail: spmn@aol.com

References

1. News Release, Office of Assistant Secretary of Defense (Public Affairs), Washington, D.C., July 13, 1994.
2. House Armed Services Committee Report 106-162, pp. 198.
3. Senate Armed Services Committee Report 106-50, pp. 340-341, May 17, 1999.

Note

1. The Software Engineering Institute has developed the Capability Maturity Model, which identifies where a company lies along a continuum of software development maturity.

CROSSTALK Wants to Hear from You

We welcome reader comments regarding CROSSTALK articles or matters pertaining to software engineering. Please send your comments and Letters to the Editor to crosstalk@stsc1.hill.af.mil or mail to

OO-ALC/TISE
Attn: CROSSTALK staff
7278 Fourth Street
Hill AFB, UT 84056-5205

Please limit letters to less than 250 words. Include your name, phone number, and e-mail address with any letter. We will withhold your name if you desire.

Best Practices – Presentation is as Important as the Issue

George Jackelen
EWA, Inc.

Internal Organization for Standardization (ISO), Institute of Electrical and Electronics Engineers (IEEE), and other organizations spend a lot of time and money documenting best practices to improve quality, reduce cost, and the like. Many of these best practices are ignored due to their complexity. Businesses need best practices that are easy to implement. In many cases the problem is not the best practices per se, but their presentation.

Introduction

Development of best practices is usually based on experiences and updating current standards. Formal best practice implementation begins with documenting these best practices. Thus, a "good" best practice is doable and beneficial (based on experience) and repeatable (based on documentation).

Best practices are usually documented/presented as standards, policies, procedures, checklists, and reports (e.g. ISO technical reports).

Presentation Problems

A major problem is how the best practice is presented. Poor presentation can result in best practices being ignored. Poor presentation problems include:

- COMPLEXITY. Does a best practice have to be presented in 50-plus pages? Is the presentation providing too much detail? Are only minimum mandatory requirements presented? Can "mandatory" details be replaced by guidelines in a separate document? Too much mandatory detail normally results in costly implementation, (e.g. the best practice is not feasible for small companies). Best practices need to define/explain what needs to be done. Providing an implementation example as an annex can be helpful. Providing mandatory requirement details about how to implement a best practice often results in the best practice not being feasible.
- UNDERSTANDABILITY. One of the beauties of Einstein's $E = MC^2$ is its simplicity and ease of understanding, without requiring knowledge of the details. Best practices need to use common terminology and definitions. Best practices must have a common-sense view to the implementers. To be considered for implementation, best practices should be clearly understandable to managerial and technical people.
- RIGIDITY. Best practices should be stated in a general fashion to allow for flexibility of implementation. In the worst case (e.g. complex best practices), best practices need to be tailorable and provide an easy method to show compliance (e.g. when certification is needed). Best practices need to allow for addition, deletion, and modification of requirements [1]. In some cases, tailoring and compliance statements can be used [2].
- APPLICABILITY. Best practices should be applicable across domains. This increases a best practice's usefulness by increasing its application to several business activities.
- EFFECTIVE WRITING. The computer age does not appear to have improved our writing skills. I have seen many documented best practices where the authors are so concerned about meeting deadlines and being technically correct that they ignore spelling, grammar, and writing style. These problems detract from the value of the best practices and may even reflect the problems we are having with software systems (e.g. if people cannot write documents well, they are prone to develop bad code) [3].

The ISO 9000 series is an example of best practices meeting the principles of the above list. The new ISO 9001: 2000 is attempting to improve its presentation over the current and previous versions [1]. By relying on quality manuals/plans to explain how organizations comply and implement an ISO 9000 standard (best practice), ISO allows for simplicity, ease of understanding, flexibility, and a wide application. The author believes this is a major reason for the worldwide acceptance of the ISO 9000 series.

Conclusion

When documenting best practices, remember to KISS (Keep It Simple, Stupid). What good is a best practice if its presentation makes the best practice too difficult to implement? Why not allow organizations the freedom to determine the best way to implement best practices? Finally, if needed, have a technical editor/writer look at your best practice (drafts and final) before distributing. ◆

About the Author

George Jackelen is project manager and analyst for two NASA IV&V projects. During his more than 30 years experience, he has performed software and hardware quality assurance for the Department of Defense (DoD) and industry and has developed and/or provided review comments on ISO, IEEE, DoD, and contractor standards, policies, and procedures.

He is also working on ISO projects to develop software life cycle standards and technical reports.

EWA Inc.
1000 Technology Drive
Fairmont, W.Va. 26554
Voice: 304-367-8252
Fax: 304-367-0775
E-mail: gjackele@ewa.com

References

1. ISO 9001: 2000 Quality Management Systems — Requirements (draft), March 1999.
2. IEEE/EIA 12207.0-1996, Industry Implementation of International Standard ISO/IEC 12207: 1995 Standard for Information Technology — Software Life Cycle Processes, March 1998.
3. May, Lorin, "The Key to Effective Writing and Coding: Quality Assurance," *CrossTalk*, May 1999.

Coming Events

Software Testing Analysis & Review (STAR) '99 West

Theme: Improving Software Testing and Quality Engineering Practices Worldwide
Dates: Nov. 1-5, 1999
Location: San Jose, Calif.
Sponsor: Software Quality Engineering
Topics: Specific ways to improve testing efforts and results. Field-proven techniques for testing client/server, object-oriented, global information infrastructure, and Internet applications. How to use test engineering to consistently achieve greater software quality. The best Internet/Web testing tools and how to use them effectively. How to lower development costs and boost productivity with test engineering.
Voice: 1-800-423-8378 or 904-278-0707
Fax: 904-278-4380
E-mail: sqeinfo@sqe.com

Managing Projects Well

Dates: Nov. 2-5, 1999
Location: Denver, Colo.
Sponsor: Quality Assurance Institute
Focus: This four-day seminar/workshop teaches you what you need to know to lead or be a member of a project team. It will also discuss the real world of projects and what they do not teach you in project management school.
Voice: 407-363-1111
Fax: 407-363-1112
Internet: http://www.qaiusa.com

Effective Methods of System Testing

Dates: Nov. 8-11, 1999
Location: Washington D.C. area
Sponsor: Quality Assurance Institute
Focus: This four-day seminar will enable you to effectively plan and execute software testing to validate that a system meets requirements.
Voice: 407-363-1111
Fax: 407-363-1112

Third International Software Quality Week Europe '99

Dates: Nov. 8-12, 1999
Location: Brussels, Belgium
Sponsor: Software Research Institute
Topic: The conference theme, "Lessons Learned," reflects the accomplishments of the past few years, and aims to see what can be learned from such efforts as the Y2K, Euro Conversion, the push for e-Commerce, and the widespread use of mature software quality processes.
Contact: Rita Bral
E-mail: bral@soft.com

24th Annual Software Engineering Workshop Call for Papers

Dates: Dec. 1-2, 1999
Location: Green Belt, Md.
Sponsor: NASA/Goddard Space Flight Center Software Engineering Laboratory, University of Maryland and Computer Sciences Corporation
Internet: http://sel.gsfc.nasa.gov/sew.htm

12th Annual Software Technology Conference

Theme: Software and Systems — Managing Risk, Complexity, Compatibility, and Change
Dates: Apr. 30-May 4, 2000
Location: Salt Lake City, Utah
Co-Sponsors: Air Force, Army, Navy, Marine Corps, Defense Information Systems Agency, Utah State University Extension
Co-hosted by: Ogden Air Logistics Center/CC, Air Force Software Technology Support Center
Contact: Dana Dovenbarger
Voice: 801-777-7411
Fax: 801-775-4932
E-mail: dana.dovenbarger@hill.af.mil

Tinker Air Force Base's Software Division Wins IEEE Award

Walt Lipke, deputy chief of Tinker's software division, accepts the IEEE Award for Software Process Achievement from Watts Humphrey.

The Oklahoma City Air Logistics Center Directorate of Aircraft Management Software Division's Test Software and Industrial Automation Branches won the IEEE Award for Software Process Achievement.

The award was announced in May at the Internal Conference on Software Engineering in Los Angeles and presented in August.

"Just to be named a finalist for this award is a major achievement," said Kelley Butler, the division's focal point for quality and process improvement. Past winners include NASA/Goddard in 1994; Raytheon, 1995; and Hughes, 1997. In 1996 and 1998, applicants' achievements were not considered significant enough to warrant giving the award.

Tinker's Test Program Set and Industrial Automation functions were selected as one of two finalists in late 1998. A review board representing the Institute of Electrical and Electronics Engineering came to Tinker in March to discuss issues and concerns it had with the information in the division's nomination package.

The review team consisted of Barry Boehm, professor at the University of Southern California's Center for Software Engineering; Victor Basili, professor at the University of Maryland's Computer Science Department; Watts Humphrey, fellow at the Software Engineering Institute (SEI); Manny Lehman, professor in the Distributed Software Engineering Section at Imperial College, London, England; and Bill Riddle, SEI.

The division received a set of five detailed questions from the IEEE evaluation team in January and prepared its responses, which were given to the review team in March. The questions were designed for respondents to clarify and expand upon points presented in the nomination package.

"The preparation for responding to the team's questions was very similar to getting ready for an essay-type exam you would have in a college class," said Walt Lipke, deputy chief of Tinker's software division.

"The first question was so involved, it took four hours to complete our response. At that point, I wondered if we would be able to complete the review in one day. But after the initial question was answered, many of the issues in the remaining questions were already answered. Only two hours were needed for the last four issues," said Lipke.

The next day, Lipke learned of the review team's unanimous recommendation that the IEEE select Tinker for the award.

In its selection, the review team cited the Test Software and Industrial Automation Branches for Tinker's outstanding record of process and product improvements. In March 1993, Tinker's software division was the first Air Force organization to achieve Level 2 from the five-level SEI Software Capability Maturity Model. In November 1996, these software functions became the first in federal service to achieve a Level 4 rating. In September 1998, these components achieved registration of their quality system. They did this by successfully passing an audit against the requirements of ISO 9001/TickIT — the software implementation of the international standard for quality systems — Model for Quality Assurance in Design, Development, Production, Installation, and Servicing.

The software improvements have provided significant customer benefits. Over the last five years, the amount of effort nominally required to design, develop, and deliver a test program set has been reduced by 37 percent, a reduction from 1,600 to 1,000 man hours. The cycle time, from the beginning of the project to completion, was reduced by 15 percent, going from 13 to 11 months; product defects have been eradicated to 99 percent.

"Accepting the IEEE Award for Software Process Achievement at the International Conference on behalf of Tinker's software division was a great thrill and honor for me," said Lipke. "Receiving the award confirms the significance of the software process improvement achievements." ◆

CrossTALK

Sponsor	*Lt. Col. Joe Jarzombek*
Publisher	*Reuel S. Alder*
Associate Publisher	*Tracy Stauder*
Managing Editor	*Kathy Gurchiek*
Associate Editor/Layout	*Heather L. King*
Associate Editor/Features	*Heather Winward*
Voice	801-775-5555 DSN 775-5555
Fax	801-777-8069 DSN 777-8069
E-mail	crosstalk@stsc1.hill.af.mil
STSC Online	http://www.stsc.hill.af.mil
CrossTalk Online	http://www.stsc.hill.af.mil/ Crosstalk/crostalk.html
ESIP Online	http://www.esip.hill.af.mil

Subscriptions: Send correspondence concerning subscriptions and changes of address to the following address:

Ogden ALC/TISE
7278 Fourth Street
Hill AFB, Utah 84056-5205
E-mail: custserv@software.hill.af.mil
Voice: 801-775-5555
Fax: 801-777-8069 DSN 777-8069

Editorial Matters: Correspondence concerning Letters to the Editor or other editorial matters should be sent to the same address listed above to the attention of *CrossTalk Editor*.

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 *Guidelines for CrossTalk Authors*, available upon request. We do 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: This can be reached on the Internet. World Wide Web access is at http://www.stsc.hill.af.mil. Call 801-777-7026 or DSN 777-7026 for assistance, or e-mail to schreifr@software.hill.af.mil.

Back Issues Available: The STSC sometimes has extra copies of back issues of *CrossTalk* available free of charge. If you would like a copy of the printed edition of this or another issue of *CrossTalk*, or would like to subscribe, please contact the customer service address listed above.

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 that will improve the quality of their software products, their efficiency in producing them, and their ability to accurately predict the cost and schedule of their delivery. *CrossTalk* is assembled, printed, and distributed by the Defense Automated Printing Service, Hill AFB, Utah 84056. *CrossTalk* is distributed without charge to individuals actively involved in the defense software development process.

Who Knows Best?

In this fast paced software industry the latest panacea for sale is called “best practices.” To jump-start an organization’s performance it seems like a very logical strategy. Find out who does what best in the industry and emulate them.

“Hey Wally, I’ve got a great idea. Let’s get Eddie Haskle to teach us how to build a pinewood derby car. He told me he won last year so you know he can build a winner.”

“I don’t know Beav, can we trust Eddie?”

“Sure Wally, I can see the trophy now. Dad’s going to be proud!”

Sounds simple — and that is where the danger lies. No matter what it is, we want it now, and we don’t want to get off the couch to get it. We have created a lethargic consumerism that’s transitioned from the corner store, to the drive-up window, to home delivery. Books, clothes, food, tickets, and even cars can be purchased on the Internet and delivered to your doorstep the next day — but at what price?

Best practices smack of that same quick and easy mentality. Adopting a best practice is not an easy process and should not be our only improvement strategy. In the steeplechase for best practices there are four questions, or hurdles, that will require research, thought, and effort.

“Wally,this will be the fastest car in the Rain Gutter Regatta.”

“Regatta? Beave, you dough head, the regatta is a boat race not a car race!”

The first hurdle: what best practice do we need? This isn’t the fashion industry; just because someone calls it a best practice does not mean we need it. Go to the best practices market with an idea of what is going to help.

“Gee Wally, we did everything Eddie said and we came in dead last.”

“I’m not sure Eddie was telling the truth about winning last year. Everyone at the derby said Stubby Wallace won three years in a row.”

The second hurdle: how do we know which practice is best? Before hooking up with the Eddie Haskles of the software market, do your research. Robert Glass in the April 1999 issue of *Communications of the ACM* investigated one of the most productive practices in the industry — inspections. Most journals, magazines, and trade shows indicate best inspection practices are founded in formal inspections — assigned roles, pre-inspection training, several reviewers, and formal meetings. Glass found that informal inspections were more effective, and two to three participants were sufficient. He warns of “… a peculiar dichotomy our field has — we laud with our hearts, not with our heads.”

“It looked so easy when Eddie was showing us how to sand pinewood, but now we are back home, I can’t sand my car evenly.”

“Yeah, maybe Dad could get a neat workshop like Mr. Haskle’s.”

The third hurdle: how do we know a best practice will work for our organization? If Ada worked for Booch does that mean it will work for you? This may be the toughest hurdle of all — the one with the water behind it. Consider available resources, scalability, and your organization’s culture when adopting a best practice. Discounting any one of these factors leads to disaster.

“Boys, I don’t mind you going to Eddie or Stumpy for ideas but the only way you are going to beat them is to come up with a better idea of your own. That, mixed with hard work, will serve you well.”

The fourth and final hurdle: are your practices better than the best? If we chase each other’s best practices, eventually innovation will die.

Our employees live and breathe the business daily and are a valuable source of creativity, ingenuity, and practicality. Cultivate their ideas.

Unlike fast, faster, and fastest — best has no connection in origin with good. You have to make that connection.

— Gary Petersen, TRI-COR Industries

Got an idea for *BackTALK*? Send an e-mail to backtalk@stsc1.hill.af.mil

Software Best Practices

Field Report

Open Forum

Departments and Announcements

CROSSTALK
Ogden ALC/TISE
7278 Fourth Street
Hill AFB, UT 84056-5205