In-the-SPIN

Newsletter of the Boston SPIN

Issue 36 April/May, 2000

Editor: Carol Pilch

Editorial

This issue of In-the-SPIN has a SPIN Perspectives column that may be very timely for many software organizations since it ties into staffing issues. After reading Rick Brenner's perspective, you may want to reconsider your strategies for accounting for staffing shortfalls. Also, this issue features another segment from Judi Brodman, the SPIN Doctor. Judi's column focuses on the benefits of peer reviews, QA, CM, etc. to the project. Also, she presents "the evolutionary SQA role," as experienced by the Xerox Corporation. And, if you missed the panel discussion "Process Maturity: Things that Work" at the April meeting, David Heimann provides the summary of that meeting.

Remember that the June meeting will feature the annual election of the SPIN Steering Committee members. If you are a Boston SPIN member, you are eligible to nominate candidates, be nominated as a candidate, and vote at the June meeting. You will be hearing about the nominating process at the May meeting.

If you're a reader of this newsletter, the Boston SPIN would greatly appreciate your feedback. The Boston SPIN, and in particular the editor, would like to know if the readers' expectations are being met. The SPIN steering committee also encourages broader participation in the content and production of the newsletter. Send letters-to-the-editor, quips, quotes, anecdotes, articles, offers to participate in the newsletter committee, and general correspondence to Carol Pilch, carol.pilch@GD-CS.COM.

Boston SPIN
Since January 1993

Software Process Improvement Network

SPONSORS:

General Dynamics Communication Systems

Raytheon Company

Edelman & Associates

We thank the Computer Science department of **UMASS- Lowell** for providing support and hosting our Web page

SPIN Perspectives

This edition's SPIN Perspective is contributed by Richard Brenner. Rick is Principal of Chaco Canyon Consulting and is an at-large member of the Boston SPIN Steering Committee.

When Is Change For a Dollar only 82¢ ?

If there are some people in your organization who have some extraordinarily rare skills, knowledge or talents, they're probably in high demand. You've tried to hire more like them, but it's next to impossible. So you time-share them. You assign them 50% to this project, 30% to a second, and 20% to a third. Oh, and if some other little thing should come up now and then, they get to do that, too. This might not be good management—it might not be your preferred way to work—but that's life.

I suggest that the costs of this arrangement could be significantly higher than most of us ever imagined. After reading this, I hope you'll look for some other way to deal with the problem. And I'll give you some ideas.

Continued on next page

IN THIS ISSUE . . .

The hidden costs of split assignments

For someone with multiple assignments, life is complex. They bear all the invisible costs of dividing their attention, and since we rarely account for these costs explicitly, they're buried, appearing only as depressed performance. Here are some examples of these invisible costs.

Shifting gears Split assignments increase the number of different activities in a typical day. Whenever you shift gears, you temporarily set aside one context, and slide another into place. This takes time—depending on the individual and the complexity of the work, it can be ten or fifteen minutes [DeMarco 87, p. 63]

Frequency of interruptions Split assignments increase the number of people you work with, which increases the chances that a phone call, a page, or an email will interrupt you—more time lost to shifting gears.

Calendar management On split assignment, your calendar is more complex. The effort you spend managing your calendar for a 50% task is about the same as for a 100% task. Moreover, the total effort you spend on calendar management increases faster than the number of assignments, because of interactions and conflicts between them.

Conflicts internal and external The conflicting priorities that you feel when you're on split assignment can also appear internally. You want to focus on each assignment as if it was your only task, but you can't. This can create a sense of conflict between your desire to perform at ever higher levels and your merely human ability. This internal conflict is a source of stress, which can lead to depressed productivity and performance.

Damage to relationships All kinds of relationships can be affected. For example, when you are on split assignment, you have two project leaders. Their conflicting priorities can lead to questions about your personal loyalties and trustworthiness. If this happens, the organization can suffer. It's almost impossible to account for this kind of cost on a spreadsheet, but it is very real.

Stress Stress arises because we fail so consistently to notice all the overhead items in this list. Somehow the gaps have to be closed, and if the person on split assignment is achievement-oriented, he or she is the one most likely to take responsibility for closing those gaps, which usually means extra hours. And extra hours can often mean shoddy work due to fatigue. For a person who takes pride in performance, this is a source of stress, which leads to lower productivity, higher rework rates, increased sick time, and schedule disruption.

Perhaps stress management training can help, but it's far better to eliminate the source of stress. Starting a stress management education program instead of reducing sources of stress is a lot like teaching pedestrians how to do self-surgery rather than building a pedestrian bridge over the freeway.

Rework rates When you're on split assignment, it's more difficult to get into flow [Csikszentmihalyi 1991]. Average rework rates are higher, which might not be so bad. But when average rework rates climb for a person whose role in a project is central—whose work affects the work of large numbers of other project team members—there can be big trouble. If their work needs rework, it might happen that the work of those who depended on the upstream results might also need revision. As a project leader, it's important to make sure that rework rates are low for people whose output is far upstream. Often, these are precisely the people who are on split assignment. Uh-oh.

Reporting People on split assignments write reports, just like everyone else, but they write one report for each partial assignment. So they spend more of their time reporting. Moreover, their reports are processed in just the same way as are the reports of a full-time team members. A team of 20 full-timers might have to deal with 20 reports per time period. But if the 20 are actually full-time-equivalents, and there are really 32 people on the team, then there are 32 reports, not 20. And correspondingly higher numbers of reminders to people who are late with reports, higher numbers of reports to read, to file, to merge together, to compare with each other, and so on.

Travel When a person on a half-time assignment travels, two things happen. First, while on travel, they become full time. Second, the costs of travel—airfare, food, and lodging—are full costs, not half costs, even for a half-time team member. The cost of sending a part-time team member off site is sometimes unrecognized during budgeting exercises—it's the same as the cost of sending a full-time member.

Continued on next page

Meeting costs The cost of a meeting is the hourly cost of each person attending multiplied by the amount of time they invested in the amount of time they invested in the meeting. But meetings are like travel. Whenever someone is attending a meeting, they're full time on the project, whether they are dedicated to the project or only part-time on the project. When you use split-project assignments, you raise the average time spent in meetings, relative to full-time assignments.

What you can do

I hope you're wondering by now how to reduce the number of people who are on split assignments. The obvious solution, hiring more people, is rarely practical. Downward pressures on headcount not only preclude this, but also may even exacerbate the problem. But there's a lot you can do that will reduce your reliance on hiring.

Encourage mentoring If you have someone with rare skills or knowledge, consider each project as an opportunity for that person to mentor one or more others. Use projects as training opportunities—propagate capability throughout your organization.

Use consultants as capability builders We most often use consultants to perform tasks directly, with little effort spent teaching our staff the skills the consultants have. Consider using consultants—both internal and external—to educate your people as they work.

Coordinate projects Coordinate projects to avoid contention for the same people. If you can schedule an activity in one project so that it doesn't overlap a similar activity in another, the people who have to work both activities can do them one at a time, rather than simultaneously.

Cancel or delay projects It might be nice to get this project done, but is it really a priority? If you can't delay or cancel an entire project, perhaps you can delay or reschedule those parts of it that create the greatest potential need to have split-time assignments.

Reuse components If two projects are building similar components, perhaps you can make the components identical. If you can, you might eliminate the contention for the time of the same people.

Buy components Perhaps you don't buy components built by other organizations, but if the alternative is splitting the time of a number of people, the factors above might enter into the build-buy decision.

Splitting people across projects can be like getting only $0.82 when you ask for change for a dollar. You start out with one FTE, and by the time you deduct all the overhead costs of splitting their time, you have much less than one FTE. The return on effort spent trying to avoid split-time project assignments can appear immediately, and it can help lower the level of frenzy in your organization.

Meeting Summary

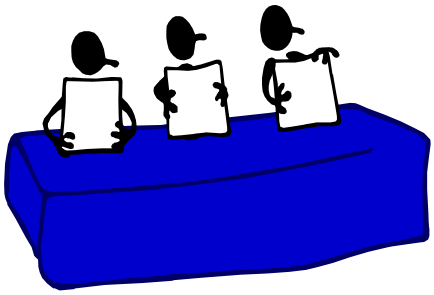
Notes from the April Meeting

Contributed by David Heimann, Comverse Network Systems

Panel: Process Maturity: Things that Work

Moderator: Donna Johnson, LOGOS International

Panelists:
Barry Foster, MatrixOne, Inc.
Dolores McCarthy, Computer Sciences Corporation
Carol Pilch, General Dynamics Communication Systems



On Tuesday, April 18, the Boston SPIN held a panel discussion on "Process Maturity: Things That Work". The panelists were from a mixture of different organizations with different perspectives: Barry Foster of MatrixOne (a commercial company), Dolores McCarthy of CSC (an on-site contractor to the Volpe National Transportation Systems Center, a non-Defense-Department government agency), and Carol Pilch of General Dynamics (a defense contractor). The moderator was Donna Johnson, an independent consultant on software process improvement.

In her introduction, Donna Johnson ranked the Key Process Areas for the Capability Maturity Model Level 2 and Level 3 in order of least-often satisfied to most-often satisfied. For Level 2 the order was

- Software Subcontract Management
- Software Quality Assurance
- Software Project Planning, Software Project Tracking & Oversight
- Software Configuration Management
- Requirements Management

Continued on next page

For Level 3 the order was

- Integrated Software Management
- Training Program
- Organization Process Definition
- Software Product Engineering
- Intergroup Coordination
- Peer Reviews
- Organization Process Focus.

Barry Foster asserted two key enablers; Senior Executive Support and Existence of A Strategic Objective (in his company's case it was Technology Leadership). The process improvements that worked were:

- Neutral Assessment of Maturity Level
- Document What We Do
- Project Post Mortems
- Brown Bag Luncheons (to share procedures)
- Electronic Process Implementation (using their own product)
- Small Steps to Success (such as starting the use of peer reviews on maintenance releases, where it's easier to use them)
- Engineering team involvement with initiative (with control and influence over the process improvement and engineering buy-in at the highest level).

Barry mentioned that the time frame for showing short-term improvement was 18-24 months. In the long-term the time frame was 24-36.

Dolores McCarthy listed the following process improvements that worked:

- Tracking requirements changes
- Documenting existing software lifecycles
- Peer reviews
- Organization charts
- Posting project schedule and updates
- Configuration management plans
- Quality assurance plans
- Consistent test plans

Her recommendations were as follows:

- Work in reality (know what "is," move towards achievable parts of "should")
- Set expectations
- Report results
- Praise
- Place software-process-improvement assets on an intranet
- Research to update assets
- Hang in for the long term
- Spread the good news
- Just-in-time training
- Make changes gradually
- Be flexible
- Stay positive
- Training (they have developed a metrics course)

As to the time frame, she mentioned that certain small improvements, such as a document template or implementing configuration management, could be done in a few weeks or months, but that a full-bore process improvement (reaching CMM Level 2 or 3) would take at least 1 1/2 years.

Carol Pilch mentioned the following as enablers:

- Demonstrated senior management commitment (more than lip service; real involvement)
- A full-time Software Engineering Process Group (SEPG -- that is treated as a project and whose members are held accountable as in a project)
- Monthly Software Status Reviews
- Early use of metrics (with a designated person to organize metrics)
- Process-tailoring guidelines
- A Software Project Management course, "our" process, taught by internal subject-matter experts
- Assessments and Improvement Plans (in response to assessment findings)
- A Software Quality Assurance group, that is independent of the projects and the SEPG.

She listed the following recommendations:

- CMM used as a roadmap
- CMM training for all levels of management
- Periodic software status reviewed by senior management
- Dedicated SEPG staffed by experienced software engineers/managers
- Independent Software Quality Assurance
- Standard software toolset with dedicated support
- Early measurement -- size, progress, quality
- In-house training provided by subject matter experts

The time frame for the Corporate Software Process Improvement activities has been a 15-year period (1985-2000) and is still continuous.

Information about Upcoming Meetings

by Johanna Rothman, Program Chair

May Meeting Announcement

Topic: Dynamic Business Planning

Speaker: John Satta

When: Tuesday, May 16, 2000. 6:30pm-8:30pm
6:30-7:00 Networking and Round Tables
7:00-7:10 Announcements
7:10-8:10 John Satta: Dynamic Business Planning
8:10-8:30 Questions and Answers

Who: Everyone (Academia, Government, Industry)

Location: General Dynamics, 77 "A" St., Needham MA.

Abstract: Today the world of business is moving at Internet speed. Now more than ever, enterprises need to react quickly and accurately. The only consistent way to do that is to connect your strategic vision to the ever-evolving environment that is your business reality. The strategy should be consciously set at the top, generally a good vantage point from which to see opportunities and obstacles. But that vision must be based on an accurate understanding of the true direction of the organization.

In this presentation we will discuss:

1 - how the planning process can and must be continual, not periodic
2 - using dynamic planning to communicate across the organization
3 - getting and keeping stakeholders involved
4 - planning for change in the face of uncertainty
5 - practical considerations and pitfalls to watch out for.

Linking process improvement to the business goals makes it possible to continue doing process improvement. In this presentation, John will explore several processes that can connect your strategic vision to the everyday business realities.

About the Speaker: John Satta is the Product Manager for the VisionTools product line from Quality Systems and Software. John started his career designing electronics for various military contractors. He then co-founded Oryx Corporation and led a small team that designed and built the then-fastest fully programmable DSP system. Later, John consulted on system design for several large military contractors on projects including the USAF F-22 advanced tactical fighter and the USN New Attack Submarine. Looking for a change of pace, John spent a year in the video wall industry, leading the development of a 12 foot by 60 foot wall for the Nasdaq Stock Market as physical incarnation of the virtual stock market.

Now as Product Planning Manager for Quality Systems and Software, John is using his experience in developing large complex systems to define and develop products that help industry successfully manage complexity in today's ever-accelerating business environment.

SPIN Roundtables: Roundtables are focused group or "birds-of-a-feather" discussions, with a facilitator, to stimulate and moderate discussion. Roundtables are held during the Networking portion of the SPIN meeting. See our web page, http://www.cs.uml.edu/Boston-SPIN to see which topics are selected for this SPIN meeting.

Directions: From Route 128 in Needham, take exit 19A onto Highland Avenue East. Take your first right by the Ground Round and take your second left onto "A" Street. General Dynamics is the last building on the right. Enter the parking lot by the General Dynamics sign and come into the building by the cafeteria entrance, which is located to the left of the main entrance. There will be a security guard at the entrance. See http://www.gd-cs.com/needham.html for directions.

Info: See our web page, http://www.cs.uml.edu/Boston-SPIN For SPIN info, contact Johanna Rothman, 781-641-4046, or jr@jrothman.com

Cancellations (including weather cancellations): We will notify the membership via email to the SPIN distribution list, post the notice on the SPIN web page, and send the cancellation announcement to Channel 7 TV and radio, WRKO AM 680 starting at 3pm.

SPIN '99-'00 Program and Speaker Schedule

Date	Speaker/Topic
June 20, 2000 @ General Dynamics	Steve Rakitin "Yellow Sticky Method of Project Scheduling"

Looking for Interesting Speakers

We are always looking for interesting speakers. If you'd like to speak at Boston SPIN, please review these criteria before sending us an abstract.

Speaker Criteria:

1. The topic must be timely, an issue of concern to our membership.
2. We want to hear about real-world topics. If you have an academic topic, we're interested in how it applies to the real world.
3. If you are interested in creating a panel, please write an abstract, and suggest at least one panelist. We can work with you to find other panelists.
4. The topic should either explain how to *do* something, or bend our brains in another direction.

Continued on next page

5. The presenter should be intimately involved with the "hows" of the thing that got done.
6. We are not interested in sales pitches.

If you have information you'd like us to hear, please send an abstract to Johanna Rothman, jr@jrothman.com. Or, contact Johanna at 781-641-4046.

We developed a speaker checklist so that none of us would have to rely on our short-term memories. Please use the checklist to prepare for your SPIN talk.

Speaker Checklist:

1. 60 days in advance of meeting deliver: 2 paragraph abstract, one paragraph bio, and picture to jr@jrothman.com

2. Within one week of meeting date: If desired, email copy of paper or overheads to heimann@world.std.com so that it is downloadable from the SPIN web page.

3. At the meeting: Speaker provides one copy of overheads to Charlie Ryan for our library.

4. Optional, but highly desired: Send a copy of overheads, paper, etc. for our web page as of the day of the meeting. If possible, provide 50-60 copies of overheads at the SPIN meeting. (The attendees really appreciate this.)

5. Optional: If you've written a book and are willing to donate it to SPIN, we'd be happy to offer the book as a door prize by conducting a free drawing.

Dear SPIN Doctor

The ''Dear SPIN Doctor column is contributed by the SPIN Doctor, Judi Brodman.

This column is for you; let's make a difference! Send your comments and questions to "Dear SPIN Doctor" at brodman@LOGOS-Intl.com or directly to the Editor. Sign them or use a "pen-name" -- I respect your confidentiality.

Is SQA the pot of gold??

Dear SPINners:

I've been reading the continuing discussions on Software Quality Assurance (SQA) in the last few issues of In-the-SPIN (Issues 32, 33, and 34). In the roundtable summary 'SQA in Small Organizations' (Issue 33), the roundtable participants listed indicators of quality problems they had seen in their organizations or on their projects – what I categorize as customer quality indicators (customer complaints, high defect rates, late software deliveries) and organizational quality indicators (significant rework activity, developer complaints, loss of staff, loss of follow-on contracts). Many of the indicators are related to quality, but, in most cases, SQA, as defined by the Software Engineering Institute's (SEI) Software Capability Maturity Model (SW-CMM), would not eliminate the root causes of these problems.

High defect rates contribute to a majority of the problems - customer complaints, developer complaints, late software deliveries, significant rework activity, loss of staff, loss of follow-on contracts. To drop the defect rate, the roundtable participants suggested initiating peer reviews for code changes during testing. I would suggest using the peer review process earlier in the lifecycle - ideally when requirements are being generated.

Statistics show that inserting peer reviews early in the software development cycle eliminates more defects than waiting to conduct peer reviews on code. But, if that's a stretch for your organization or project at this time, start peer reviewing your design concepts and then peer review your code before testing.

The buddy system, as suggested, helps identify flaws but many small software organizations are now successfully using a more formal peer review process with great success. This alternative peer review process consists of the following steps:

- the item to be peer reviewed is distributed to the selected reviewers
- the item is reviewed by the reviewers during a specified period of time
- the defects found by the reviewers are documented and returned to the author.

The missing step is the meeting of the reviewers to collect and discuss the defects found. It has been documented that eliminating the meeting with the reviewers has little or no effect on the numbers of defects found. If you want to lower that high defect rate, institute peer reviews on your projects.

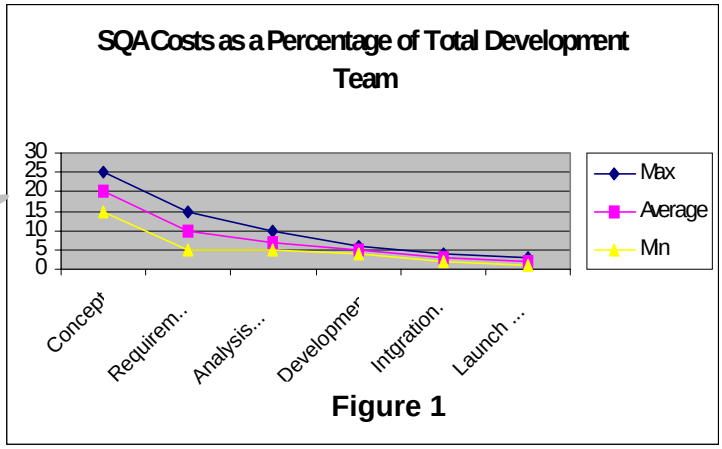


Figure 1

Continued on next page

This peer review process can take place without an SQA individual on the project and can take place on a project without 'convincing management to take action in the area of quality.' For the peer review process, SQA confirms that the item to be reviewed was sent out to all the listed reviewers with sufficient time for review; that the reviewers spent the allotted time reviewing the item; that all reviewers provided their comments to the author; and that the author followed up on all problems reported. If SQA does exist in your organization, it can provide the checks and balances but the developers still need to do the work of reducing the defects!

Peer reviews are only one solution to the problems indicated by the roundtable. Other solutions lie in the institutionalization of the following activities: estimating project effort using well-defined estimation techniques, scheduling and allocating resources using realistic effort estimates, developing requirements based on customer input, and controlling changes to the established software requirements. Traditionally, these activities need to be in place before the SQA individual can perform audits and reviews on the activities and their products.

Later in this column, I plan to discuss an interesting evolutionary SQA role, which many of you may find very interesting. First, let's address an SQA question from "Energized Jim".

'Energized Jim' asks: *How does adding the reviews, QA, CM, etc. impact a typical project and how much additional effort is required on average?*

Watts Humphrey has written that the SQA effort should be around 3% of the organization's resources. In a software development group of 100 developers, 3% is probably feasible but in a small software organization of about 20 or fewer developers, 3% doesn't work. In my experience, most small organizations (20 software developers) have one full-time SQA individual (5% of the organization) who coordinates and performs SQA activities and anywhere between 1 - 4 part-time SQA people (1 - 5% of the organization) assigned to perform project SQA.

As a result of my effort to find more actual SQA data other than my experience and Watts' recommendations, I came across an interesting presentation by John Kostecki of Xerox Corporation called 'Evolutionary Software Quality Assurance'. This presentation provided interesting data on the resources involved in implementing SQA as a percentage of the total project development team. John was kind enough to let me share this data with you. Figure 1 shows the SQA percentages of the total team for each software development lifecycle phase. Percentages vary from an average of 20% of the development team during the Conception Phase of the project to 3% of the development team during the Launch/Maintenance Phase.

But before you take these percentages away, we need to discuss the evolutionary SQA role that I talked about earlier because the structure of this role impacts the percentages that John showed in Figure 1. In most organizations today, the SQA individual plans their own activities based on the project schedule, and as the project matures, reviews project's activities, audits work products, and provides management with notice of any deviations from plans, standards or procedures.

Xerox decided to take the words expressed at the beginning of the CMM SQA KPA to heart - "*The SQA individual works with the software project during the early stages to establish plans, standards, and procedures that will add value to the software project and satisfy the constraints of the project and the organization's policies. By participating in establishing the plans, standards, and procedures, the SQA individual helps ensure the plans, standards, and procedures fit the project's needs and verifies that they will be usable for performing reviews and audits throughout the software lifecycle."* Based on these words, they developed an evolutionary SQA role, a role that combines SEPG activities with SQA activities, a role that matures and changes as the project and organization mature and change.

The Xerox SQA individual performs the following roles:

- Process coach,
- Process monitor,
- Work product auditor,
- Statistical process controller,
- Process metrics collector and analyzer, and
- Defect prevention guru.

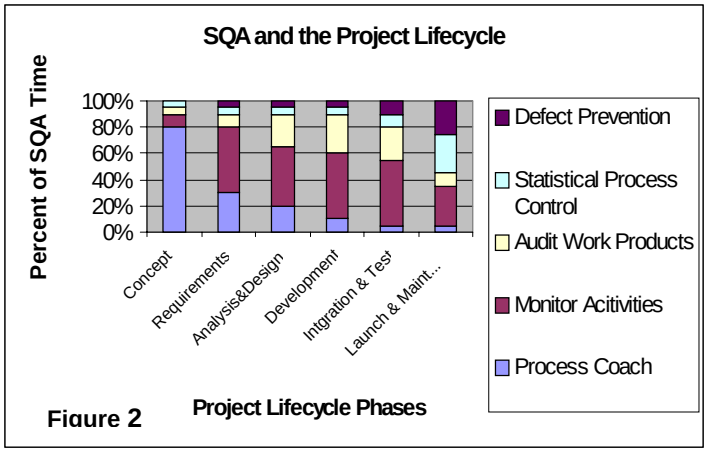


Figure 2

Figure 2 shows how the role of the SQA individual changes as the project changes and matures. In the Conception Phase, the SQA individual spends his/her time as follows: 80% as a Process coach, 10% as Process monitor, 5% as work product auditor, 5% as Statistical process controller, and 0% defect prevention guru. In the Launch/Maintenance Phase, the time split is as follows: 5% as Process coach, 30% as Process monitor, 10% as work product auditor, 30% as Statistical process controller, and 25% defect prevention guru.

Continued on next page

Let's take a closer look at the SQA role on a project during the Level 1-2 transition or when the organization is immature

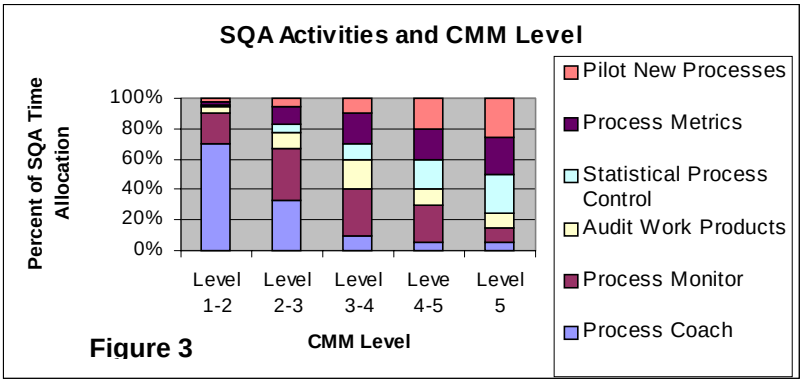


Figure 3

(refer to Figure 3). In John's words, "In organizations starting SPI, the entire organization is Level 1 and, by definition, has no written process. It is therefore fruitless to engage in auditing as a primary function." He has found that by using some consultative advice on the projects they can move toward building a repeatable process. He states "At that point (when a process exists), audits can be performed to insure that the processes and procedures are being followed".

During the Concept Phase, 80% of the SQA individual's time is utilized as a process coach. The SQA individual, at this time, is functioning as the Software Engineering Process Group (SEPG) on the project - providing the expertise on the CMM and process definition as it applies to the project. As John explains it, the SQA individual participates in the following activities:

- identifying the process needs of the project,
- documenting the project's processes, procedures, measures and standards,
- developing and/or documenting the software life cycle processes, and
- Cross-pollinating 'best practices' between projects.

The evolutionary SQA role allows the SQA individual to work as part of the project team right from the start, coaching and helping the team develop and document the process activities and artifacts. As the project matures, the role of SQA changes as shown by the percentage of time allocated to auditing and reviews and defect prevention (Figure 2). The SQA individual takes on the role of process coach in the beginning of the project and therefore is very familiar with the process and products making his/her role on the project less intrusive as the project mature.

As the organization matures to Level 2 and has an established SPI program in place, Figure 3 shows SQA concentrating more on audits. John adds, "BUT they will also coach the project leads and engineers on how to do self-auditing. The objective is to move more of the mundane procedure compliance checks to the working level. He/she moves gradually into a role of auditing as the engineering teams are executing the process checkpoints." Although the SQA individual's role as Process Coach appears to diminish, SQA is now working in the SEPG to help establish organizational processes. Having worked at the project level as the process coach as the organization matured, the SQA individual brings to the Level 3 effort a whole host of best practices as well as worst practices.

Once the organization reaches level 3, the SQA team members are assigned to each of the process improvement teams where they review new procedures and standards to determine if procedures and standards can be audited. Since the organization is still encouraging projects to perform self-audits, the standards and procedures have to lend themselves to self-audits.

At level 4, the SQA individual becomes Software Quality Manager (SQM) and as such take on a more proactive role in the organization. It is at this point, John says, that the SQMs develop and manage the project's Software Quality Plan. John says that "formal audits are performed at set times, but have a broader scope than when the organization was Level 1. Again, compliance is measured against the organization's self-auditing capability."

At level 5, John says that "Defect Prevention becomes a key process area that requires the attention of SQA (SQM). SQA should manage this set of activities even at lower levels. At Level 5, it is an organizational business practice."

The evolutionary SQA role that Xerox has developed initiates the SEPG role early in the improvement program (Level 1 - Level 2 transition stage) making SQA audits and reviews a normal fallout of that role. This evolutionary role enlarges the SQA role from a pure audit and review role to a process role, of which SQA is a part, that changes and matures as the organization and projects change and mature. Small organizations struggle with a standalone SQA role on a project because that SQA resource is viewed as a non-value added role to the project. The evolutionary SQA role appears to solve this problem - it provides a process champion who proves the checks and balances on the project.

I'd like to hear from those of you who have initiated the self-auditing technique. Let me know how you did the self-auditing and how it worked in your organization!

Happy Spring! - the SPIN Doctor

This column is for you; let's make a difference! Send your comments and questions on this column or any other subject to "Dear SPIN Doctor" at brodman@LOGOS-Intl.com or directly to the Editor. Sign them or use a "pen-name" - I respect your confidentiality.

Boston SPIN

The Boston SPIN is a forum for the free and open exchange of software process improvement experiences and ideas. Meetings are usually held on third Tuesdays, September - June. Boston SPIN welcomes volunteers and sponsors. There is no charge to attend the meetings.

For more information about our programs and events contact:

Charlie Ryan
ESC/DIJ (Building 1624, Room 2NE15)
5 Eglin Street
Hanscom AFB, MA 01731-2100
Telephone: (781) 377-8324
Email: ryan@sei.cmu.edu

For information about SPINs in general including ***HOW TO START A SPIN*** contact:

Dawna Baird of SEI (412) 268-5539, dbaird@sei.cmu.edu, http://www.sei.cmu.edu/collaborating/spins/spins.start.html.

IN THE SPIN is available on our Web page:

http://www.cs.uml.edu/Boston-SPIN.

TO RECEIVE NOTIFICATION OF NEW IN-THE-SPIN ISSUES and Boston SPIN specific notices send email addressed to withall@mediaone.net

We have 2 separate email lists: one for this newsletter and one containing announcements that we receive from other process organizations and forward out.

IF YOU WANT TO ADD YOURSELF TO THE ANNOUNCEMENTS LIST send email to ryan@sei.cmu.edu.

Send letters-to-the-editor, and general correspondence to Carol Pilch, carol.pilch@GD-CS.COM.

Send job postings to heimann@world.std.com.

Back issues and other information about Boston SPIN can be found at our WEB HOME PAGE:
http://www.cs.uml.edu/Boston-SPIN/