

In-the-SPIN

Newsletter of the

Boston SPIN

Our tenth year!

Software Process Improvement Network

Issue 49 September 2002

Editor: Sheila Lynch

From the Editor

Welcome Back!

What a busy summer we have had, helping us to keep our mind off the heat! First, let me introduce the officers you elected at the June meeting.

Chair	Barb Purchia
Vice Chair	Richard Green
Secretary	Michael Brother
Treasurer	Ron Kay
At Large	Rick Brenner
	John Britis

Those officers met over the summer and selected the following committee chairpersons.

Membership	Jim Withall
Program	Barry Mirrer
Roundtables	Caroline Starita
Webmaster	David Heimann
Newsletter	Sheila Lynch



Together, these officers and chairpersons comprise the voting members of the 2002-2003 SPIN Steering Committee. In addition, the Steering Committee appoints other ad-hoc committee chairs and welcomes volunteers for many of our activities. Please let us know if you have an interest in becoming more active in your local SPIN.

We are also presently in the process of constructing a new Web site which we hope to have on-line shortly.

Sheila Lynch, Editor, *In-the-SPIN*,
Email: salynch@mitre.org

From the Chair

Hello Boston SPINners,

We hope you enjoyed your summer. This year marks the Tenth Anniversary of the Boston SPIN! With fall rapidly approaching, your Steering Committee has already been working hard to make this coming year the best ever.

Barry Mirrer, our Program Chair, is gathering a list of outstanding speakers for our 2002-2003 program. So far, we have Michael Mah for our opening meeting, September 17, 2002 at MITRE and commitments from Larry Constantine, Sam Guckenheimer, Jim Highsmith, Capers Jones, and Johanna Rothman.

Caroline Starita, our Roundtable Chair, is generating a delectable selection of roundtable topics that will be aligned with our SPIN speakers' topics. In addition, we will continue our ongoing roundtables: the Software Testing Forum, the SPIN Book Club, the Software Process Improvement Forum, and the Hiring Initiative.

Until further notice, the official domain name and Web site location of the Boston SPIN Web site will be: <http://www.cs.uml.edu/Boston-SPIN>. Dave Heimann, our SPIN Webmaster, has been doing a terrific job maintaining the Web site.

Sheila Lynch, our *In-the-SPIN* Newsletter Chair, has collected informative articles for this edition. And, Judi Brodman, our SPIN Doctor, is working on a special Tenth Anniversary *In-the-SPIN* edition. If any of you have any special memories (words and photos) of the Boston SPIN (meetings, friendships, job opportunities, and so on) over the past 10 years, Sheila (salynch@mitre.org) and Judi (brodman@logos-intl.com) would love to hear from you.

SEPG 2003, the Software Engineering Process Group Conference, will take place in Boston, Feb. 24-27. This is the premier international conference and exposition for software process professionals. If you want more information on this conference, go to <http://www.sei.cmu.edu/sepg>.

Welcome back to the Boston SPIN and we look forward to seeing you on September 17 and throughout the year.

Barbara Purchia
Boston SPIN Chair

IN THIS ISSUE . . .

From the Editor.....	1
From the Chair.....	1
Speaker Spotlight - Adaptive Software Management.....	2
Feature Article - Climbing out of Technical Debt.....	2
June Meeting Synopsis.....	4
September Meeting Preview.....	5
SPIN Information.....	7
Upcoming Meetings.....	7

Speaker Spotlight

ADAPTIVE SOFTWARE MANAGEMENT — STRATEGIES DURING AN ECONOMIC DOWNTURN

by Michael Mah

From Cutter Consortium Web site, dated April 10, 2002, <http://www.cutter.com/freestuff/bitadvisor.html>, Business IT Strategies, Email advisor, reprinted with the permission of Michael Mah, Senior Consultant, Cutter Consortium.

Throughout much of the economic boom times of the mid- to late 1990s, developing software at Internet speed was the rage. Most organizations employed strategies intensely focused on time to market, oftentimes without regard to cost.

Today the situation is quite different. In what feels like the blink of an eye, software managers have had to adapt to an environment of significant cost reduction. In many cases, the method in which this plays out is through budget and staff constraints. The pendulum has swung to the other side. Rather than staffing up to a level required to build an application within a given time frame, many managers today are constrained to running a project with whatever staff resources are available to them.



In his writings on software agility, Cutter Consortium Senior Consultant Jim Highsmith has expressed the idea that in today's environment, pressures to deliver software quickly and to change quickly (and often) are compelling reasons to rethink traditional software engineering practices. The same concept applies to software management practices. Managers today need to respond with agility on project decisions — setting budgets, delivery dates, and, most important, how much functionality they commit to — within those budget and staff constraints.

Unfortunately, many managers I speak with every day rely only on manual methods to establish targets and project commitments. Some even admit to using the "wet finger in the wind" technique to estimate project time and scope within a given budget and staff size. The problem with this is that multimillion-dollar decisions are often at stake, frequently in an environment where the dynamics of change are occurring faster than ever before.

Traditional manual approaches are too heavy and cumbersome in this environment. They take too long to get answers that executives need. As a result, inertia takes over, with many projects being set into motion by default with whatever is handed to them, without examining the cost reduction, schedule, and scope tradeoffs upfront.

Here is where many of today's commercial software estimation tools help. Their purpose is to provide agility and speed to help managers explore such tradeoff scenarios prior

to, and during the project's critical stages. If cost reduction is the order of the day, then teams can examine the potential schedule tradeoff, and assess whether the expected completion of the desired scope will be within the target deadline.

If the resultant analysis suggests that the deadline is at too high of a risk, the team might have to do a triage analysis to evaluate whether the scope needs to be cut back and renegotiated. This is "time-boxing" in action. The next step would be to employ effective negotiating techniques to come to an agreement on whether what can be delivered will meet the needs of the organization or its intended market.

Project managers who do this successfully avoid the last and least desirable scenario — where inertia sets up the project to constrained staff/budget, under an impossible deadline, while taking on too much in the way of project scope. This is the so-called "Yourdon Death March" project. They are better able to steer clear of this situation, having better odds at avoiding multimillion-dollar overruns or those situations where software is deployed with too many bugs. Having the agility to make better decisions early is an insurance policy against these scenarios, which can be disastrous during an economic climate of containing costs. Not only are cost overruns difficult to absorb in this environment, but software deployed with high defects can drive maintenance and field support costs through the roof.

Finally, as the economic climate improves, it may turn out that budgetary and staff constraints are eventually relaxed. Again, managers who are able to respond adaptively can explore new scenarios where they "dial in" a strategy to ramp back up. These tradeoffs can be assessed to evaluate how the throughput of the organization might improve, deploying more software in a shorter time frame. They can react more quickly to change when the time comes, and adjust their strategy accordingly to meet the marketplace.

In the end, what we are striving to achieve is an ability to make better decisions proactively, rather than being in a situation of reacting to fire drills on projects that drift into chaos. This adaptability is about software managers being more empowered to steer their projects more effectively, with better information and to stay ahead of the curve.

Feature Article

Climbing out of Technical Debt



© 2002 Johanna Rothman

Have you ever had a conversation like this one?

Vice President: In the last release, you were able to bring in the release date by over a month by cutting the testing. Do that again, ok?

Project Manager: Sorry, boss, we can't do that. For the past three releases we've shortchanged the design work and the testing. We can't cut any time from this

project. In fact, it's time to re-architect the system, redesign it, and do the testing we've put off.

Vice President: NO WAY!

Project Manager: Hey, that's just the way it is. We haven't taken the time to finish this product yet, and our shortcuts have come home to roost. Our technical debt is too high to work this release the way we've worked the others.

Technical debt, as defined by my colleague Dave Smith, is the debt a company "owes" to a product they persisted in shipping in an incomplete or unstable condition. This is a situation referred to in Hunt and Thomas's book *The Pragmatic Programmer* as "software rot" or by Gerald M. Weinberg in *Quality Software Management, Volume 4* as "design maintenance debt." The software isn't necessarily rotten—the product is incomplete.

As the technical debt increases, the load on the customer support staff becomes overwhelming, and the developers have trouble adding or changing system features. When developers are stuck on new development and support is overloaded, you face difficult choices about what to do now: have developers support the current product; ignore the current product and continue with development; some combination of support and development; or stop developing and fix what you've already got.



Project managers, development managers and test managers often can see this coming. They know there is a window of opportunity to fix the product before the technical debt overwhelms the company's ability to do new product development.

Look for these signs of technical debt:

You ship a product and then have to put out a point release before it's even left the CD duplication house.

Testing time increases disproportionately to the development time.

The developers tell you that part of the product needs to be re-architected, because the current architecture doesn't support the current requirements, never mind the additional requirements.

Developers refuse to touch a part of the product saying, "No one but Fred can touch that. I know Fred left three years ago, but no one else can work on it, because we don't understand it."

Developers spend more time supporting current customers by solving customer problems, fixing defects, and answering customer questions, than they spend developing new product.

You hire more testers than you have developers, and you're still not sure you've tested the product.

Your developers threaten to leave because all they do is "maintenance".

You ship the product not because it's ready, but because the developers' families have abducted them to go on vacation, or the developers are too exhausted to continue the crunch mode you've been in.

You stop testing because you don't want to find more defects.

Your cost to fix a defect continues to increase, from release to release. (If you'd like to see a picture of this dynamic, check out Weinberg's *Quality Software Management, Volume 4*, reference below.

Recognize your technical debt, and start planning what to do. Before you can plan, you need to know a little about how you came to have this technical debt problem. These questions may help you understand how your product acquired its technical debt:

How do you staff projects? Do you assign enough people at the time they are needed, or are some positions never assigned or assigned late to a project? If you never assign enough testers, or you don't have an architect, you will incur technical debt. Don't blindly accept people starting on your projects late, because you can't make up time in a project. Have people start when they're supposed to start, or recalculate the project schedule.

How do you design your projects? Do you first architect the entire product for its entire lifetime? Do you iterate on the design for several projects? If you don't architect the product, and update the architecture as necessary, or if you don't do a high level design before starting to code, you will not have a congruous product. One way to recognize an incongruous product is when you have multiple ways of doing the same thing. At one organization, the developers couldn't agree on how to open and save data files. Since they couldn't agree, every time a developer wanted to open or save their data file, they had their own open and save functions. When they moved from a command line interface (CLI) to a GUI interface, they had no easy way to integrate the open and save functionality.

How do you test the product? Do you have a way to start testing the product at the beginning of the project? Do you

only have manual tests through the graphical user interface? If you start limited testing late, you're guaranteeing yourself technical debt. You have many opportunities to test early and often: requirements inspections, design reviews, code inspection or peer review, unit tests, daily builds and smoke tests (tests to verify the daily build works at least a little bit), peer review of fixes in addition to system test. Decide what makes sense for your environment, and add test capability as early in the project as possible.



How do you organize your projects? Do you have functional managers who hand off the product from the analysts to the developers to the testers to the manufacturing folks? If so, no one has the same view of what the product should be. At a minimum, assign an overall project manager or program manager, so someone is pulling the entire project together. Projects without project managers are highly risky, and in my

experience, always fail. In addition, projects without a sponsor or champion have a difficult time finishing.

How do you know what done means? Do you have some objective criteria to know when the project is complete? If not, you're more likely to release an incomplete product. Develop release criteria for every project, so that everyone on the project, and the project sponsor understands what your company will receive for its project money.



How do you run your projects? Do you stick to only one type of lifecycle and one process? You may need to choose other lifecycles and/or processes to avoid future technical debt. Many project managers are comfortable with a particular lifecycle and/or process, and it can be difficult for those project managers to switch to a different lifecycle for a project. If you're used to a waterfall lifecycle, you'll have to learn to plan for an iterative or incremental lifecycle differently. And, you'll have to monitor your projects differently.

If you've changed your lifecycle, you may need to change your process. What you do and when you do it will make the lifecycle successful and keep down technical debt. If you've never planned when to do inspections and reviews, start with something small, such as peer review of defect fixes during system test. I've found that a nightly build and smoke test helps find defects faster, and helps everyone realize when the product is not ready to ship.

Although it feels as if you're moving faster when you take shortcuts on a project, in reality, your technical debt prevents you from making project progress. You might be able to shortchange your product on one or two releases, but at some point your product technical debt is like credit card debt — if you don't pay it off in a planned way, the debt buries you.

Observe and measure your products, to know when you should act to reduce or avoid technical debt.

For more information, see:

- ❖ Hunt, Andrew and David Thomas, *The Pragmatic Programmer*. Addison—Wesley, Boston, MA 1999.
- ❖ Weinberg, Gerald M., *Quality Software Management, Volume 4*. Dorset House, New York, 1998



June Meeting Synopsis

Bookclub

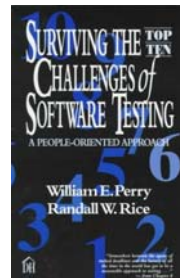
Surviving the Top Ten Challenges of Software Testing: A People-Oriented Approach

by William E. Perry and Randall W. Rice

Facilitated by Barbara Purchia, Rational Software and Paul Piper

In June, the Software Testing Forum joined tables with the Book Club to review Surviving the Top Ten Challenges of Software Testing: A People-Oriented Approach by William E. Perry and Randall W. Rice. We picked four of our favorite challenges to discuss and understand if we had encountered any of these challenges, and how we had handled them to see what worked and what didn't.

Challenge Number 2: Fighting a Lose-Lose Situation – This situation occurs when testers have to report on the product. If they report that the software is not ready for production, they are seen as roadblocks to progress. If they certify that the software is ready for production and problems appear after release, the testers are blamed for not testing the product sufficiently.



Several people in the group faced this situation and had encountered it multiple times. One suggestion was to have strong project management communicate to management. Another suggestion was to document all known problems in release notes (or at least problems that may be seen or were reported by customers.) It was also recommended that the testers provide a heads up to support about the issues and to also use support to help test the products. One person suggested that management should sign off on what's not being tested. In some cases, defect metrics were used to determine percent test coverage, percent test cases completed, percent features tested, and percent of functions tested.

Several people had developed or used test plans; however, progress was not tracked to the test plans. One person mentioned that the problem was "us versus them." It was suggested that this be turned into a win/win situation by engaging the people who will turn around the adversarial perspective and convincing upper management that the goal is for everyone to look good.

Challenge Number 10: Getting Trained in Testing – Testing is a professional discipline, requiring unique skills that are anything but intuitive. Without training, testers are not equipped to meet the rigors of testing, especially in technically difficult situations.

Several people had taken certification courses (ASQ, CSTE) and recommended that you identify the course that most addresses the skill set you wish to acquire. Some of the

September Meeting

Main Event

Guest Speaker – Michael Mah

Software Estimation and Negotiation — "Changing the Game" in a Down Economy

First, there was "Internet Speed". Now - a down economy. For the software manager, both create extreme challenges for managers and teams when it comes to software estimation and project negotiation. Deadlines are more compressed, within ever-tighter budgets and staff constraints.



Managers and their teams can either play the game, saying yes to "death march" project scenarios, hoping for a way out of the dilemma later on, or find a way to change the game. Michael's talk will address why playing the game is so hard, and what might be done to break the cycle. What role do metrics play? How can teams "sanity check" their promises and negotiate better scenarios? How does one discuss options and trade-offs when there appear to be no alternatives to impossible deadlines and budgets? As a manager, what can you do to break the cycle and prevent runaway projects?

Building upon ideas from the bestseller, *Getting to YES*, this session explores resolving complex business issues using a framework to manage the tension between competition and cooperation. We will identify:

- ❖ Alternatives to "Positional" Bargaining
- ❖ How to Determine What's "Fair" About a Project Estimate
- ❖ Techniques to Create Multiple Project Options
- ❖ Enhancing Communication and Dealing with Difficult People and Situations

Michael Mah is Managing Partner of QSM Associates Inc., and has served as past editor of *IT Metrics Strategies* with the Cutter Consortium, an industry think tank in Arlington, MA. He's a frequent industry speaker on software measurement and estimation, with a special emphasis on negotiation for in-house and outsourced applications development. Michael was also the keynote speaker with Watts Humphrey at the SEPG 2002 conference, where he played "Charlie Rose" to Watts Humphrey as the "celebrity" in a totally ad-hoc one-on-one interview on the state of the software industry.

Michael's recent work merges research on software metrics and management with negotiation techniques originally developed in the field of dispute resolution. He's worked with a wide range of corporate clients such as Rockwell, Intel, Compaq, JP Morgan, Merrill Lynch, BellSouth, Sprint, and others. He has a degree in electrical engineering from Tufts University in Medford MA, with his training on negotiation and mediation through the Program on Negotiation at Harvard

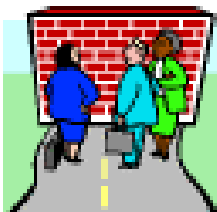
courses require self study and some also provide training before the certification exam.

We discussed having developers testing their own work and most people identified the following problems with this approach:

- ❖ It's difficult to be objective about your own work
- ❖ There isn't enough time to write tests and to debug tests and the code
- ❖ Programmers don't want to test
- ❖ It's like letting the fox into the chicken coop

Several people had found that their development skills were obsolete and began working in testing. There was a strong feeling that management does not value testing skills and therefore doesn't want to invest in training.

Challenge Number 4: Testing What's Thrown Over the Wall - Traditionally, developers and testers don't talk to each other.



When developers sometimes dump the product on the testers, they assume that the testers will find all the defects. This assumption leads to poor or non-existent testing by the developers.

Lots of people had this experience. At one company, QA was given the product in the morning and the product had to ship the next day! One company sold one of their divisions in order to solve the quality problem. Some people had to reverse engineer the requirements from the product.

One person observed: "Testing will not improve the quality of the product. It either has it or it doesn't." No matter when the product ships, it will have bugs.

Challenge Number 9: Building Relationships with Developers – When testing is viewed as something done by someone else, two sides are automatically created. However, testing is a shared responsibility and everyone in the organization should be involved in testing. This challenge is focused on moving from the "us versus them" mindset to the interdependent teamwork approach.

This issue occurs independent of whether QA is centralized or decentralized. The discussion focused on working together and communicating. Have QA participate in inspections and technical reviews.

We could have continued discussing the remaining six challenges. Read the book and if there's interest, we'll discuss them at a future meeting.



Law School (<http://www.pon.harvard.edu>) and the Radcliffe Institute for Advanced Study

Michael can be reached at QSM Associates, Inc., Clock Tower Business Park, 75 South Church Street, Suite 600, Pittsfield MA 01201, (413) 499-0988. Web site: (<http://www.qsma.com>).

Roundtables

September Book Club – “Getting Past No: Negotiating Your Way From Confrontation to Cooperation”

by William Ury

Boston SPIN Book Club Facilitator:
John Britis



Last season the book club looked at William Ury's *Getting to Yes*, a book that provides guidance for setting up win-win negotiations. Many who participated wondered, “but what do you do if you are faced by an opponent who is not interested in a win-win?”

This month's book club will discuss another of Ury's books that addresses that issue, *Getting Past No*. This book is a must read for just about anyone who interacts with people. William Ury has written a very practical, easily read, guide and process that anyone can use right from the start. Not just a 'business guide', his five-step process is easily applied to everyday situations and with practice is a foundation for much larger negotiations. The principles are well defined via a five step process that is demonstrated through examples that are fresh, relevant, understood by common association. Gearing concepts through example gives the reader a sense of self-mastery without having to memorize lists. The framework builds upon itself with frequent review of previously introduced terms. From business to interpersonal communication, this book has something for everyone.

This is also an 'easy read' at fewer than 200 pages, the chapters are subtitled by concept and reference is easy. This book's conciseness is deceptive. The concepts expressed are profound. For example, I cannot count the number of times I have been able to use of the concept of BATNA (best alternative to a negotiated agreement, i.e. what you do if the negotiations fail).

Read it, use it, and enjoy a more satisfying approach to negotiation strategy. Even alone, each of the five steps provides a valuable resource to summon upon when communications/negotiations are not moving smoothly.

Process Improvement Forum

Leader: Judi Brodman

Project disasters — Can process avert them?

We all have seen the statistics on the numbers of projects that fail but never the reasons why these projects fail. Many of us

have worked on disastrous projects as a developer or a manager. This month's PI Roundtable will address some of the reasons project disasters happen and how disasters could have been averted.

Bring your ideas on why your projects have not been successful — examples:

- ❖ estimates that are generated once and never revisited during the project life cycle therefore setting false expectations;
- ❖ deviations in schedule, resources, requirements that could have been seen early and corrected therefore averting disaster;
- ❖ incomplete requirements that caused a loss of schedule time but could have been ironed out in RAD/JAD sessions;
- ❖ too many process requirements on a small project causing developers to be buried under paperwork!

Bring your project's problems to the table and let's see if more or less process could have prevented the disaster!

Jobs Forum

Leaders: Paul Edelman and Karl Heinemann

The Jobs Forum is an open discussion of job needs and leads. Facilitators are Paul Edelman and Karl Heinemann. Both bring a wealth of industry experience to the forum, from two very different perspectives.

Anyone interested in finding a new job or that has opportunities to offer can expect a spirited discussion.

Managing Project Tradeoffs

Facilitator: Johanna Rothman



When you're managing a project or are a lead in the project, what tradeoffs do you make? How do you decide what you can trade off and when? How do you talk about the project tradeoffs with the project sponsors or stakeholders?

Join us for a discussion of project tradeoffs, how you make them, and who makes them.

Improving Software Estimates

Facilitator: Donna Johnson

Generating software estimates at the beginning of a project is one of the most challenging tasks of software professionals. There are variables that make the task even more difficult, e.g., end dates and budgets set by management.

Come discuss methods that have worked to generate more accurate estimates and ways to overcome impossible end dates, as well as other problems associated with estimating.



SPIN Information

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. Additional information about the Boston SPIN can be found at our Web home page: <http://www.cs.uml.edu/Boston-SPIN/bos-SPIN.html>.

For more information about our programs and events contact Barry Mirrer, Program Chair, bmirrer@alum.mit.edu.

Order Your Book Club Books through SPIN

Boston SPIN is bringing you more convenience for your book club selections. You can now order Book Club selections and your purchase benefits SPIN. We receive a percentage of every purchase made by linking to Amazon.com from our site.

Cancellations (including weather)

Starting at 3pm, we'll notify you via email to the SPIN distribution list, we'll post the notice on the SPIN web page, and we'll send the cancellation announcement to Channel 7 TV and WRKO AM 680.

SPIN Meeting Location

Boston SPIN meetings are held at The MITRE Corporation in Bedford.

Please be aware that MITRE has advised us that, due to increased security concerns, you will need a Picture ID for admission to the SPIN meetings. We encourage you to leave all carrying bags, backpacks, and briefcases behind (i.e., in your car).



Otherwise, you should be prepared to have these opened and inspected upon arrival.

MITRE's campus is located at 202 Burlington Road (Route 62), Bedford. SPIN meetings are held in the 'S' building. Directions can be found on our Web site: <http://www.cs.uml.edu/Boston-SPIN/bos-SPIN.html> or on The MITRE Corporation Web site.

Sponsors

The following organizations/individuals support the Boston SPIN:

- ❖ The MITRE Corporation <http://www.mitre.org/>
- ❖ Raytheon Company <http://www.raytheon.com/>
- ❖ Edelman & Associates <http://www.edeltech.com/>
- ❖ Quality Search <http://qualsearch.com/>
- ❖ UMASS – Lowell <http://www.cs.uml.edu/>

Email Lists

To receive Boston SPIN specific notices, send an email to:

- ❖ Jim Withall, Boston SPIN membership chair, at withall@mediaone.net

Future Programs

We welcome your suggestions for future Boston SPIN programs. We are always looking for interesting speakers. If you'd like to speak at Boston SPIN, please review the criteria specified on the Boston SPIN Web site before sending an abstract to:

- ❖ Barry Mirrer, Boston SPIN program chair, at bmirrer@alum.mit.edu.

Newsletter Call for Articles

The *In-the-SPIN Newsletter* is always in need of new and interesting articles dealing with Process Improvement, software development methodologies, Project Management and other related subjects that may be of interest to our readership. Please send any general correspondence or articles that you would like to have considered for publication in the Newsletter to:

- ❖ Sheila Lynch, Co-editor of In-the-SPIN, at salynch@mitre.org

Back issues of the *In-the-SPIN Newsletter* can be found on the Boston SPIN Web site: <http://www.cs.uml.edu/Boston-SPIN/bos-SPIN.html>.

Upcoming Meetings

Barry Mirrer is still finalizing our program for the year. As speakers are confirmed, we will let you know. For the purposes of reserving dates on your calendar, our tentative 2002—2003 schedule follows.

Oct. 15, '02	Larry Constantine	Improving Software Usability through Better Process
Nov. 19, '02	Capers Jones	Software Quality in 2002 – a survey of the state of the art
Dec. 17, '02	Unmesh Gundewar	Staff-less SQA
Jan. 21, '03	Johanna Rothman	TBD
Feb. 18, '03	Sam Guckenheimer	Patterns
Feb. 24-27, '03	SEPG Conference	Boston
Mar. 18, '03	Robin Goldsmith	Non-CMM Software Process Improvement
Apr. 30, '03 (note date)	Jim Highsmith	Adaptive Software Process
May 20, '03	TBD	
June 17, '03	TBD	