

---

## SYNOPSIS OF ICSQ

---

### CONTINUED

---

#### ENCOURAGING PROCESS COMPLIANCE THROUGH AUTOMATED WORKFLOW MANAGEMENT

**Scott P. Duncan**  
**SoftQual Consulting**

Scott presented the challenges of a financial services company transitioning from a paper-based system to automated workflow management. The company's existing systems were not well controlled, characterized by private libraries of inconsistent documentation, poor version control, and inability to determine project deliverable status.

An automated workflow management system was deemed necessary to guarantee that everyone could tell project/deliverable status.

Problems developed early. There was an initial lack of agreement on the requirements and piloting/training. There was outright resistance to the new methodology. Usability issues were given inadequate trials. Performance issues such as server capacity, remote site communication, marginal desktops, and database corruption were common.

Senior management "invisibility" impeded product assurance. There was no authority over use of the methodology, no authority over configuration control changes, and internal audit findings were not directed to the responsible management. There was overwhelming evidence of lack of client approvals, lack of reviews and walkthroughs, technical management approval, and testing documentation.

Predictably, project compliance reporting data showed that the percentage of projects with delivery and technical errors ranged from 42% to 85%. Given these data, managers began tracking projects by requesting specific training, and greater responsiveness to the data. The automated system provided a platform to launch improvement and allow staff to manage workflow, deliverables, and project status.

*Michael Kress is an associate technical fellow for the Boeing Commercial Airplane Group in Seattle. He is a member of the Boeing Technical Fellowship Subcouncil. He is a Senior member of ASQ and chair-elect of ASQ's Software Division.*

---

## FROM THE REGIONS

---

### REGION 4, CHRIS FITZGIBBON

---

Preparations continue for the 12th International Conference on Software Quality (12 ICSQ) that will be hosted by the Ottawa Valley Section 407 this October 28-30. This will be only the second time the ICSQ is held outside the United States, and given the strong U.S. dollar, it will be a very affordable conference for many.

Several ASQ sections throughout Region 4 are regularly offering courses on software quality and CSQE refresher courses. Some sections are trying to determine the level of demand for software quality courses. If you are interested in participating as a student or instructor, contact your local ASQ education chair or me.

As mentioned in my introductory column in the last edition of this newsletter, one of my objectives is to establish a network of contacts representing each ASQ section throughout Canada (Region 4). I am still searching for Software Division members

who are interested in being local points of contact for the following Canadian ASQ sections: New Brunswick, Windsor, Saskatchewan, London, Montreal, Kitchener, Nova Scotia, and Hamilton. If you are interested, please let me know ([chris@orioncanada.com](mailto:chris@orioncanada.com)).

There are several software quality groups meeting regularly throughout Region 4. The Ottawa Software Quality Association (OSQA) holds events on the last Thursday of each month. A schedule of events and a subscription to the OSQA mailing list is available from their Web site: [www.osqa.org](http://www.osqa.org). Some recent topics include: Web Testing, Web Metrics, Functional Points Analysis, Software Inspections, and Regression Testing. Similar topics are being discussed by the IEEE/ASQ Discussion Group for Software Quality in Calgary. The group meets every two weeks from September through May. A list of upcoming events and speakers is available through their Web site [www.software-quality.ab.ca](http://www.software-quality.ab.ca). There are also several active Software Process Improvement Network (SPIN) groups throughout Region 4. A list of SPINs in Canada is available at [www.sei.cmu.edu/collaborating/spins/spins.html](http://www.sei.cmu.edu/collaborating/spins/spins.html). And, the International Council on Systems Engineering ([www.incose.org](http://www.incose.org)) has a chapter in Vancouver and emerging/start-up chapters in Toronto and Montreal.

If there are software quality events in Region 4 that would be of interest to other ASQ Software Division members, let me know.

---

### REGION 10, DAVID WALKER

---

There are a number of activities taking place in our region right now.

The Ann Arbor Software Quality Professionals met February 19 over dinner at Azure at 6:30 pm. On March 19, Nancy Poma discussed ASQ's Certified Software Quality Engineer program. Send e-mail to Lynne Rago [lynnrargo@mediaone.net](mailto:lynnrargo@mediaone.net) for more information about future meetings.

The Ann Arbor IT Zone at <http://www.annarboritzone.org> is offering a number of presentations and discussions on software quality engineering.

Planning has begun for Michigan Quality 2002 in the October time frame. If you have ideas for this year, please contact Nancy Poma [nmpoma@mediaone.net](mailto:nmpoma@mediaone.net).

If you do not get periodic e-mail messages from me, please call ASQ at 800-248-1946 to give them your correct e-mail address and tell them you want to receive messages from your regional councilor.

---

## SPI FOR PRODUCTIVITY AND COST CUTTING

---

BY MARK PAULK

---

A question I have heard repeatedly in recent months is whether the Software CMM® can be used for cutting costs. The implication behind the question usually is that "quality" is the target of the CMM users, but our business objective is different, therefore the CMM is not appropriate for our needs.

This is a general question that can be asked of any model or standard for process improvement. I believe that an emphasis on "Big Q" quality, which includes factors other than just defect density, is an integral part of any successful process

improvement program. Software process improvement, whether CMM-based or not, should focus on achieving the business objectives of the organization. If those organizations have productivity or cost cutting as their top priority, then models such as the CMM can provide a useful framework for achieving that business objective.

Unfortunately few of the process improvement models and standards available provide direct guidance on the implications of such an objective. The classic Deming chain reaction suggests that higher quality lead to less rework and shorter cycle times, which in turn leads to more satisfied customers. How to achieve this result and improve productivity and cost as software quality attributes is left as an exercise for the reader.

Consider the following observations, inspired by the Software CMM.

- *Requirements Management.* Accept the fact up front that customer requirements will change. Even if customers know exactly what they want today (a rare occurrence!), the operational environment will frequently change, and the product must evolve. Cut costs by proactively accepting change from the beginning. This is the heart of agile methodologies such as Extreme Programming (XP), which “embrace change” as part of their disciplined process.
- *Software Project Planning.* Pick an evolutionary or incremental life cycle. Actively involve the customer and end user (who may be different entities) with evolving the requirements. One of the crucial success factors for XP and similar agile processes is building the product in small increments and keeping a shippable version ready. Remember that 80% of the functionality will probably come from 20% of the work. If you want to minimize cost, be prepared to declare success between the 20 and 100% points. Once the critical functionality is in place, ask whether the remaining features are important and useful or just gold plating. Realistic plans may indicate the project will take much longer than the customer finds acceptable — but ignorance doesn’t solve the problem, it just exacerbates it. An incremental life cycle provides much better control of the project and an opportunity for mutually beneficial negotiation.
- *Software Project Tracking and Oversight.* Keep an eye on rework. If errors are caught early in the life cycle, it’s a lot cheaper to fix them. Particularly for this kind of life cycle, make sure the requirements you’re working on first are really the most important ones to address first.
- *Peer Reviews.* Install some form of peer review for requirements analysis, design, code, and testing—the payback will make it worthwhile. Inspections are undoubtedly the most effective form of peer review, but they require a significant amount of infrastructure in the form of training, bug tracking tools, etc. XP advocates pair programming, which can be considered a form of ongoing peer review. This is a “pay me now or pay me (more) later” decision.
- *Software Quality Assurance.* Product assurance can be embedded in your process, if you have a well-defined process (this is a typical high maturity strategy), but process assurance is a worthwhile overhead expense. Consciously verifying that the project is doing what it said it would do prevents nasty, last-minute surprises...which can be quite costly.
- *Software Configuration Management.* Investing in basic SCM tools is fundamental. Some of the SCM functions, such as

change control, can be embedded in the development process. For example, if you’re using the XP strategy of frequent small increments being delivered to the customer based on “stories” (a.k.a. user scenarios), then the customer may be acting as the de facto change control board...and the choice of life cycle drives this part of the project’s plan for SCM.

- *Training Program.* Give people the skills they need to do their work. If you think training is too costly, just wait until you pay the rework bill for the work done by unqualified people.
- *Integrated Software Management.* Don’t reinvent the wheel. Take advantage of tools and resources built by the organization and/or other projects. This is the process side of “design for reuse.” Reusable components are cheaper in the long run.
- *Software Product Engineering.* Make tradeoff decisions for cost vs. quality that make sense in your business environment. The Software CMM does not say that all defects have to be fixed; you do have to decide how you are going to resolve defect issues. Deciding to ship a product with known defects may be a legitimate business decision, but it should be an informed decision that factors in both short- and long-term impacts.
- *Intergroup Coordination.* When commitments aren’t met, there is always a cost. Making and tracking commitments, and negotiating acceptable compromises when problems arise is much more cost effective than “letting the squeaky wheel get the grease.”

The above observations suggest how the CMM can be used as part of an improvement program oriented toward cutting costs. Cost cutting frequently involves other steps, such as layoffs and salary reductions, that may be counter-productive in the long

(cont. on p. 16)

## Software Quality Systems Software Process Improvement Software Testing

Providing practical and effective help with:

- Software Quality Assurance
- Quality Goals & Plans
- Process Improvement (using SEI CMM)
- Software Project Management
- Software Testing
- Root Cause Analysis
- Software Metrics
- Software Inspections (peer reviews)
- Software Reliability

## Software Quality First

Jessee Ring—Principal

510-915-2353 Fax: 510-573-7464

sqalst@attbi.com

---

# SPI

---

## CONTINUED

---

term. Implementing processes that factor in cost as a quality attribute seems a much more proactive strategy.

® Capability Maturity Model and CMM are registered with the U.S. Patent and Trademark Office.

---

# SOFTWARE QUALITY ENGINEERING QUIZ

---

BY LINDA WESTFALL

---

Whether you are preparing for the Certified Software Quality Engineer (CSQE) examination or just testing your knowledge of software quality engineering, why don't you sit back and let your brain do its thing. The answers can be found on p. 18 if you need a helping hand.

Note: The items in this quiz are NOT from past CSQE examinations NOR were they created as part of CSQE exam development process.

- Which of the following are defect detection techniques?
  - Requirements elicitation
  - Peer reviews
  - Structural testing
  - Architectural design
  - III only
  - II and III only
  - I, III, and IV only
  - I, II, III, and IV
- Which of the following is an example of a reliability requirement?
  - The software shall operate in both a Windows and UNIX environment
  - The software shall round all monetary calculations to the nearest 100th of a penny
  - The software shall experience no more than an average of three minutes of outage per site per year
  - The software shall print an error message to the operational reporting printer whenever a data communication interrupt is detected
- A software module has proved to be extremely error prone during both system test and field use. A decision has been made to remove this module from the product and replace it with a module that has been redesigned from scratch. This is an example of:
  - reengineering.
  - reverse engineering.
  - retirement.
  - reuse.
- The above table shows the actual status (Status), the scheduled status (Schedule), the actual cost in effort days (Actual) and the budgeted cost in effort days (Budget) of a five-task project. What is the BCWP to date for this project?
  - 25 days
  - 26 days
  - 30 days
  - 31 days
- You are conducting an analysis to determine if there is a correlation between the Cyclomatic complexity of a module and the number of field reported defects for that module. Which of the following would be best to use for this analysis?
  - Histogram
  - Pareto diagram
  - Scatter diagram
  - Run chart
- Which of the following levels of testing focuses on testing the internal structure and functions of the individual parts that make up a software product?
  - Unit testing
  - Integration testing
  - System testing
  - Acceptance testing
- The current software release uses features available only on the current operating system and will not run on previous versions of the operating system. This is an example of:
  - Reengineering.
  - Concurrency.
  - Reliability.
  - Dependency.

(Answers on p. 18)

---

# CRAWLING THROUGH THE WEB

---

BY SUE CARROLL

---

We are pleased to announce that PDF files for back issues of the newsletters will be available on the <http://www.asq-software.org> Web page. Our next challenge is to make them available for searching. If you have any expertise in this area or have time to index these files, please contact me.

Our Web redesign project is moving along—many thanks to the committee members Rufus Turpin, Frank Bonk, Scott Duncan, Joel Glazer, Dave Walker, Theresa Hunt, and Linda Westfall with ASQ support from Pablo Baez.

Have you been to the ICSQ (<http://www.icsq.org>) site? Linda Westfall is managing that site—please let her know what you think about it. Go there to find the latest information about 12ICSQ in Ottawa.

Have you been to <http://www.asq.org/pub/sqp/>? That is the Web site for the *Software Quality Professional* journal. Each issue, one article is posted to the Web along with all the resource reviews and other items from the journal. Once a year the full text of all articles is on the Web. New issues are posted in March, June, September, and December. The journal is sponsored by the Software Division. Be sure to check it out.

Let me hear from you if you have suggestions on any aspect of

	Task A	Task B	Task C	Task D	Task E
Status	Done	Done	Done	Not Started	Done
Schedule	Done	Done	Not Done	Not Done	Done
Budget	5 days	13 days	4 days	7 days	8 days
Actual	6 days	11 days	6 days	—	8 days