

# Software

Q U A L I T Y

## VISION:

To be the leading authority and recognized champion on issues related to software quality.

## MISSION:

The Software Division is the community of individuals and organizations committed to: Seeking and applying technologies, concepts, and techniques to improve the quality of software products, processes, and services; growing their professional skills; and building a stronger level of professionalism in that community.

Check out the Software Division Web site at [www.asq-software.org/](http://www.asq-software.org/)

The contents of this publication are protected by the copyright of the American Society for Quality.

Interested in reproducing this information? See ASQ's guidelines for using copyrighted material.

<http://www.asq.org/join/about/copyright/permission>



Software  
Division

## IS THERE HOPE FOR SOFTWARE QUALITY?

BY DENNIS J. FRAILEY

### Abstract

Two hundred software quality improvement plans at organizations representing a cross section of the U.S. economy show the tenuous state of software quality. This paper examines what is going on in the trenches, as reported by working professionals trying to make a difference. The results are often disappointing but there is also evidence that quality improvement is being accomplished at companies that want it.

### Introduction

At Raytheon Corporation I've observed a steadily increasing awareness of software quality and significant advances in quality engineering processes. Much of what I've learned has been incorporated into two software engineering courses I teach at Southern Methodist University on an adjunct (part-time) basis: Software Project Management and Software Measurement and Quality Engineering. These courses have a diverse (mostly working professional) clientele owing to their availability via distance education. Enrollments vary from year to year but I typically have about 100 students per year in the management course and half that in the quality course. In each course the student is required to prepare a major paper or plan that includes components directly related to software quality. In the Management course the student prepares

a software development plan, which includes a risk management plan, a measurement plan, and a quality assurance plan. The main Measurement and Quality course assignment is a quality improvement plan that features a value-added analysis, a cost-of-quality analysis, and an assessment of the best way to improve software quality at the student's place of work. The courses also require smaller student tasks, such as an assignment in the quality course to identify an example in their company where quality problems are hidden or ignored and to assess why this is happening.

Over the past 10 years I've retained copies of the work turned in by these students as well as my correspondence with many of them. My files include over 300 development plans, about 200 quality improvement plans, and about 300 reports on ignored quality problems. [At least 150 distinct companies are represented, including some of the largest and smallest corporations in the United States, mainly in commercial, government, and research sectors.] Some students continue to correspond well after they've completed the courses. I've heard tales of woe, tales of delight, and many things in-between. This paper discusses some of the patterns I've seen and some of the changes in attitudes over the last decade.

(cont. on p. 4)

# CHAIR'S CORNER

BY MICHAEL P. KRESS

## ISO 9001:2000: The Process Approach

The ISO 9001:2000 auditor needs to change his/her focus. The change to the 2000 version promotes a paradigm shift in the approach to managing quality and to auditing. The process approach says that the organization should define itself in terms of a network of processes as opposed to a string of disjoint activities designed to satisfy a set "shall" statements. Auditors are now being taught to depart from the "checklist" mentality and embrace the "objectives" mentality. "Are objectives being met" rather than "Are all 'shall statements' being satisfied" is the key shift. Why is the "objectives" approach better?

One of the reasons it is better is that an organization can be satisfying the "shalls" without meeting the objective. A classic example is filling out corrective action report (CAR) templates, assigning CAR numbers, throwing them into a database, building some pretty bar or pie charts, and then returning to creating more CARS. The process elements in CAR generation may be followed, but the objective of identifying root cause and genuinely solving the problem or a similar problem from recurring may be ignored.

The new ISO is intended to ensure the CAR's get managed and managed well. It purports to assure that management looks at the CAR data, recognizes problems, and acts to enable solutions. Auditors need to foster the same mentality.

You are an auditor and have just completed your supplier evaluation and are summarizing your findings in preparation for your supplier exit-briefing. Before summarizing your findings you should ask yourself how the findings compromise the organization's ability to meet the 8 Quality Management Principles of ISO 9000:2000 on which the ISO 9000 family of quality management system standards are based. These principles provide a framework for systematically and consistently implementing and maintaining a management system designed for continuous performance improvement while addressing the needs of all interested parties. The questions following each principle help you formulate and state, at a high level, the findings and conclusions of the audit in terms of the principles. These questions are not intended to add additional scope to the audit, but to help focus both the auditor and the supplier on the correction of meaningful findings, and avoid reporting trivial isolated escapes that do not threaten these objectives.

1. **Customer Focus**—Organizations depend on their customers and therefore should understand current and future customer needs, should meet customer requirements, and strive to exceed customer expectations.

Is management truly focused on customer expectations?

Are customer inquiries monitored, logged, and tracked to ensure timely responses and resolution of customer issues?



Is there frequent dialogue with customers to ensure effective communication?

2. **Leadership**—Leaders establish unity of purpose and direction of the organization. They should create and maintain the internal environment in which people can become fully involved in achieving the organization's objectives.

Are managers really leading?

Are the managers functioning as coaches; motivating their people?

Are managers regularly communicating with their people?

Are managers listening to their people who interface with customers?

3. **Involvement of People**—People at all levels are the essence of an organization and their full involvement enables their abilities to be used for the organization's benefit.

Are all personnel included in award and recognition events with the customer?

Is teamwork and initiative recognized and rewarded?

Is top management visible to and involved with the rank and file?

Does top management communicate goals and challenges to all teams?

4. **Process approach**—A desired result is achieved more efficiently when activities and related resources are managed as a process.

Are all major activities defined with expected inputs and outputs?

Are process parameters defined?

Are requirements for signatures, deadlines, templates to be used, quantities and standards to be followed specified?

Are the processes effective in achieving the required results?

5. **System approach to management**—Identifying, understanding and managing interrelated processes as a system contributes to the organization's effectiveness and efficiency in achieving its objectives.

Are processes integrated seamlessly?

Do outputs from one process flow efficiently to the next process?

Are process metrics used to ensure process integrity and efficiency?

6. **Continual improvement**—Continual improvement of the organization's overall performance should be a permanent objective of the organization.

Are measurements used to establish baseline performance or quality?

Is management truly using these measures to manage improvement opportunities?

Are measures truly useful or are measures mere formalities?

7. **Factual approach to decision making**—Effective decisions are based on the analysis of data and information.

Are decisions made based on facts and data?

Is an adequate level of management required to review and act on these data?

8. **Mutually beneficial supplier relationships**—An organization and its suppliers are interdependent and a mutually beneficial relationship enhances the ability of both to create value.

Is the organization open and honest with its suppliers, employees, and customers?

Is Boeing, for example, as a customer, helpful or merely critical of the supplier?

Findings should be presented in a constructive fashion. Instead of reporting findings bluntly and adversarially, the auditor should endeavor to explain how one or more of the findings is unhelpful to meeting these objectives. The following example illustrates the blunt and adversarial approach vs. a more mutually beneficial approach.

FINDING: Para 5.5.3 Internal Communication; Para 6.3 Infrastructure:

*“Supplier management is frequently unavailable to personnel.”*

A better way of reporting this might be:

*“Supplier management appears to split time between several sites, making it difficult for personnel to communicate, receive guidance, and obtain authorizations, and answer customer queries. Steps should be taken to provide better on-site availability of management. Timely communication is key to effective management of the software QA function and to customer satisfaction. The current situation is not consistent with ISO principles #2 and #3, and the objectives of ISO 9001:2000, 5.5.3 and 6.3.*

While these principles apply to the quality system in general, those of us in the software trade can benefit from them as well. The fundamental strategy does not change. If we successfully make this transition to the process-and-objectives focused approach, we will have more efficient, more effective, and longer lasting quality system improvements leading to greater profitability for all parties.

# IS THERE HOPE FOR SOFTWARE QUALITY?

## CONTINUED

### Interest in Software Quality

My perception is that students have grown steadily more interested in software quality over the past decade. In support of this I offer three forms of evidence. In the management course the students are asked to rank nine topics by level of interest (Table 1). These are not all the topics covered in the course, but they are topics on which I can vary the degree of coverage to fit student preferences, to some extent.

Using a scale of 0-5, with ties permitted, the students tell me where their interests lie. Over the years (Figure 1), software quality has slowly increased from being one of the least liked topics to one of slightly lower than average interest and its absolute score has risen from about 2.8 to around 3.7 today. Student comments have been consistent with this - in the early years quality was viewed as a necessary evil by many of the students, but today it tends to be seen in a more favorable light. This is a course aimed at potential software planners and managers. Considering the relative importance of the other topics, it would seem that software quality has made a significant advance in student interest. There has also been an increased interest in quality-related topics in student papers, projects, and plans.

A second measure of the students' interest in quality is course enrollment. Average enrollment in the required management course has doubled in 10 years, whereas enrollment in the elective quality course has tripled. The management course is known to require a lot of work and yet over half the students who take it sign up later for the quality course, which is known to present more of the same.

- Software Lifecycle Models
- Software Process
- Software Maturity Models
- Software Estimating
- Software Risk Management
- Software Planning
- Software Planning Details
- Software Configuration Management
- Software Quality

Table 1. Nine Flexible Topics in the Software Management Course

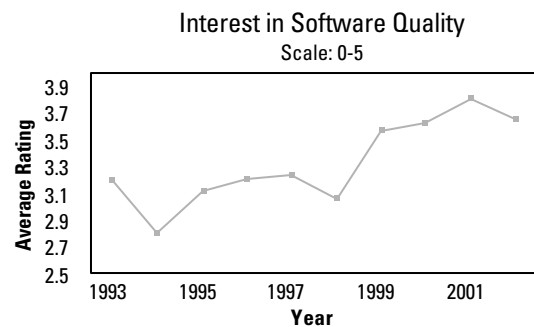


Figure 1. Student Interest in Software Quality

(cont. on p. 4)

---

# IS THERE HOPE FOR SOFTWARE QUALITY?

## CONTINUED

---

A third measure of student interest is the choices the students make in the quality course. Although this was not an option in the first few offerings, a quality improvement plan has been an optional student project for the past six years. Every year more and more students choose that option over others that do not require assessment of their own work organizations. Today, about nine students out of 10 choose the quality improvement plan option, even though it requires them to investigate their work environment, analyze quality problems, find their root causes, perform a cost of quality analysis, and make a convincing case to their management that their improvements will have a positive impact on the bottom line. Furthermore, the proposed improvements often show a remarkably sophisticated understanding of the subtler principles of quality assurance and quality engineering. All of this convinces me that more and more practitioners in the software community have a sincere interest in quality and understand the basics of how to improve it.

### 1. Tales of Woe

The quality course uses Weinberg [1992] as a primary textbook. One of the early course assignments is to document a quality problem found in the student's company and show how it fits one of the patterns of denial identified so well by Weinberg. Specifically, they are to find a case where the organization hides or denies a quality problem and discuss how it fits a recognized pattern. The students tend to enjoy this assignment and their reports range from the hilarious to the heartbreaking. Among the more memorable tales where I could not provide actual quotes because of confidentiality requests:

- The case where "value added" is officially defined as whatever the boss wants.
- The officer in an [unnamed to protect the innocent] branch of the armed services who stated that "whenever I introduce an idea from your class to my commanding officer he moves me to a desk farther from his office."
- The student who said, on his course critique and later in a personal interview, "All of this quality stuff is interesting in theory, but I can't believe anybody actually *does* it."
- The outstanding software developer who got so frustrated with poor quality he switched to the job of chief test engineer - and then his well-conceived plan for reducing defect levels was rejected because he was "only a test engineer."
- Countless cases where there was never enough time or money to spend on improvements but plenty of money to spend on rework.
- The student working on a government contract [the specific level and name of the government shall be anonymous] who pointed out that according to the accounting rules in his situation you can charge rework to the government but they won't pay for the "overhead" costs of reducing rework before the fact. In other words, you make more money by producing bad software and the accounting rules designed to save the taxpayers money actually encourage such waste. Here is a typical quote:

- "I saw inefficiencies I couldn't believe continued on a daily basis. My snooping and informing had no positive effects, in fact my boss did not appreciate hearing about my discoveries and avoided me."

These examples are discouraging enough, but things get worse when student quality improvement plans are examined. Each student's plan includes a case for action, backed up by data. The plan may, optionally, be presented to the student's management. The student also has an optional "comment on the project" section, which frequently offers good insight. Here are some facts of note, based on a survey of 25 randomly chosen quality improvement plans:

- 20% of the students provided their actual company name and did not require that the report remain confidential.
- 56% of the students provided their actual company name but insisted that the report remain confidential. (As a result of this, I was not able to provide verbatim quotes for many of the more interesting cases.)
- The remaining 24% of the students were required by their company to disguise the identity of their company and occasionally were required to make more comprehensive disguises before they were allowed to turn the plan in for grading. Some must even use noncompany e-mail addresses to communicate with me so I cannot tell where they work.
- Roughly half of the students indicated in their comments that they didn't dare actually show the plan to their managers.
- About one student in 10 reported being penalized for uncovering quality problems or proposing improvements. This has become such a concern that I must warn students not to risk their careers and must explain that they have the option of not showing the plan to anyone else. In one case reported to me in a later e-mail, the division vice-president praised the plan but the direct supervisor took it out on the individual thereafter, since the supervisor had originally seen the plan as a challenge to her competence. [My assessment of this student's plan was that it was well conceived and relatively diplomatic in focusing on process issues rather than individuals.]
- One student was so discouraged by the negative career impact of proposing improvements that he was convinced quality improvement is impossible in practice. He informed me in a later e-mail that from now on he is going to simply go to work, please the boss, and take out his frustrations by working in his garden.

### The culprits

These comments may lead one to conclude that the main objections to quality improvement come from higher-level managers. Certainly there are plenty of examples where managers don't know how to manage software projects very well or where corporate incentives for managers preclude effective focus on quality problems. However, as discussed below, I find that technical staff are equally responsible for resisting quality improvement.

- Returning to management I would cite three forms of incentive as the most counterproductive used in modern organizations:
- Rewarding people for being busy rather than rewarding them for being productive. Few companies even know how to measure genuine productivity.
- Rewarding "firefighters" but not fire preventers and planners.

Too often this is part of the company's culture and is perceived as a reason why the company achieved success in the first place (Weinberg calls these "creation myths").

- Rewarding managers for getting their individual department's jobs done cheaper or faster, disregarding the harmful effects on the overall business process. (These people need to read Goldratt [1984] or Hammer [1993]. But then who has time to read when your projects are all in so much trouble!)

Numerous other management problems were reported, including:

- The "testing is easy" attitude - putting the newest and least experienced staff on testing assignments, resulting in high levels of customer-detected defects.
- The "quality is a clerical task" view - putting the least experienced staff into quality assurance positions, where they are routinely dismissed by more experienced software developers. This often perpetuates the myth that quality assurance is a waste of time and money.
- The "our process is excellent and we follow it consistently" illusion, typically found in organizations that have overly prescriptive processes that are rarely followed.
- The "it's only maintenance" designation - applied to organizations that spend most of their time implementing new functionality but are not expected to follow a process.
- The "invisible SEPG" syndrome - the organization has a process improvement organization that is never seen by those actually doing software. Presumably it is off in an ivory tower, writing process documentation.

The "if we don't record accurate/actual time we can get free labor" perception—failing to accurately report time or failure to record overtime, thus failing to get accurate data that would permit better estimating and more accurate measurement of process costs (hence facilitating process improvement).

Yet when I tally the results of the many student comments, I conclude that equally intense and intransigent objections to quality improvement come from within the technical community. The existing software developers in many situations are unmoving in their desire to preserve the status quo - one in which they are perceived as heroes, putting out the many fires that may have been created in the first place by their own poor processes and mismanaged development techniques. This is not intentional sabotage - it stems from an honest belief that their methods are the best, perhaps because it's all they've known. And too often software developers aren't rewarded or motivated by efficiency, productivity, or customer satisfaction. They are motivated by the fun of writing and debugging software!

The excuses given are classic:

- "The requirements change too much for us to spend time controlling them."
- "Our concept of the system will change too much to worry about design defects."
- "It's faster to fix bugs than to inspect for them."
- "The customer will let us know if we have any serious problems."

Other commonly reported opinions from the technical community include:

- "We don't have a process"
- "Collecting data is too hard, so we don't bother"
- "Configuration management isn't necessary. It just adds overhead."
- "Our kind of software is so difficult that it's impossible to get all of the bugs out of it."

Some of the responsibility may also rest with customers. Time after time my students have pointed out situations where customers are unhappy with quality but not enough to take decisive action about it. Software customers continue to tolerate software failings they would not accept in other products. For example, one student told me of a customer who complained bitterly about software quality but was unwilling to accept a requirements change control protocol that might have helped reduce the problem.

### Good News

Not all is woe and despair. Among the more pleasant aspects of my teaching experience have been the many students who report progress, support, and noteworthy success in applying quality engineering techniques. I'm pleased to report that it works when people believe it will work and get the proper support. Among the happier examples I can cite:

- The individual whose improvements were so well received that she was promoted to a higher position and is now teaching these same principles within her company to other organizations.
- The company that was so impressed by the improvements made by two of my students that it sent nearly 40 employees to take the same courses.
- The student who introduced me to his boss at a conference, and whose boss proceeded to praise the student's accomplishments and thank me for the insights I had given him.
- A steady stream of e-mails from ex-students reporting slow but steady progress in quality improvement.

Students who report applying quality engineering techniques successfully sometimes have glowing reports. For example,

*"The most astonishing realization in doing this project was just the enormous amount of money that is wasted because of poor quality. And then how instituting only a few simple changes can bring nearly all those dollars down to the bottom line."*

For the majority of my students' companies, quality "assurance" consists mainly of quality control (i.e., testing and inspecting). More comprehensive techniques such as statistical process control and root-cause analysis are rarely practiced. But a number of successful techniques are used. The approaches that work tend to fall into four categories:

- Actually measuring things instead of relying on opinion and memory. For example, one student wrote, *"I found that many of my original guesses as to the problems that existed in my organization could be substantiated with data. In fact, as I was defining the current process and performing value-added analysis I came up with additional ideas for improvement.... All of this just confirmed to me that taking an analytical, data-driven method to improve product quality is a good idea."*

---

# IS THERE HOPE FOR SOFTWARE QUALITY?

## CONTINUED

---

- Inspections and walkthroughs that are facilitated, for which the participants are trained, and where good performance is rewarded.
- Dealing with human psychology. For example, making it a matter of pride and professionalism that one's software can meet the most rigorous tests by the most nit-picky of independent evaluators.
- Reward systems that create incentives for process improvement and genuine productivity (such as rewards for reduced rework) rather than systems that focus on meeting specific performance numbers.

### Summary

Ten years of teaching software management and software quality to a broad range of working professionals has provided a degree of insight into what's going on in the software development community. There's plenty of evidence that quality improvement can and does work. Unfortunately, there's even more evidence that institutions remain highly effective at resisting genuine improvement - and at stifling the energy of those who propose improvements or point out quality problems. I'm impressed by the depth of understanding and the heartfelt desire to improve exhibited by many of the working professionals who've taken my courses.

What makes the difference between the organization that improves and the organization that refuses to do so? If I could identify one factor it would be the customer. Whether it be a defense contractor or commercial company, whether the product must be reliable or safe or supportable or whatever, when the customer insists on quality the customer is more likely to get it. But the customer does not always ask or even know what to ask for. Customers may lack the power even if they have the concern. For example, monopolies tend to show less interest in customer concerns. And many customers simply don't realize that they ought to complain. One student spoke of his company's niche product that had no competition and where they issued monthly software updates to fix problems. These updates had to be installed in hundreds of hand-held devices at each customer's site. The company happily provided this service as part of a lucrative maintenance contract. Customers were so happy to have the product at all and the concept of a computerized solution was so new to this application domain that there were few complaints.

I see the solution in education of the customer, management, and technical staff. And for this I appeal to software engineering educators and to software quality professionals. Effective education can lead to an appreciation that quality improvement is possible, that professional quality engineering techniques are available, and that they do not necessarily add cost to the product (and may well reduce cost). Alignment of customers with the internal adherents of quality may be the most important aspect of this because customers have the strongest leverage.

### References

- Deming, W. Edwards. 1982 *Out of the Crisis*, MIT Press.
- Goldratt, Eliyahu M. & Jeff Cox. 1984 *The Goal*, North River Press, ISBN 0-88427-061-0.
- Hammer, Michael and James Champy. 1993 *Reengineering the Corporation — A Manifesto for Business Revolution*, Harper Business Press, ISBN 0-88730-640-3.
- Weinberg, Gerald M. 1992 *Quality Software Management, Volume 1, Systems Thinking*, Dorset House, New York, and ISBN: 0-932633-22-6.

### About the Author

**Dr. Dennis J. Frailey** is a principal fellow at Raytheon Company, an adjunct professor at Southern Methodist University, and an instructor at UCLA Extension and the University of Texas Software Quality Institute. He's been a software developer since 1962 and is widely recognized as a speaker and educator in the fields of software productivity, software measurement, software cycle time improvement, and software project management. Dr. Frailey has been a keynote speaker at several prominent conferences. He can be reached by e-mail at [Frailey@engr.smu.edu](mailto:Frailey@engr.smu.edu), or by phone at 972-344-8366.

---

## DINNER AND SOFTWARE

---

The **Baltimore Section** is planning an April 20 dinner meeting dedicated to software. The pre-dinner tutorial topic will be "Software Audits," and the after dinner main topic will be "Software Testing." \$20 with reservation, \$25 without. Location to be announced. For more information e-mail [asq0502@yahoo.com](mailto:asq0502@yahoo.com).

---

# SYNOPSIS OF 2003 FAA NATIONAL SOFTWARE CONFERENCE RENO, NEVADA • SEPT. 16-19, 2003

---

BY MIKE KRESS

---

## Foreword:

*This year's event covered more than 40 papers outlining the advancement of processes, methodologies, and regulations within the FAA for airborne software. The proceedings cover more than 500 pages and are three inches thick, and are available from the author. This synopsis summarizes some of these papers deemed most relevant to the certification, configuration control, and conformity processes for software.*

## Aircraft Certification Software and Complex Electronic Hardware Program Management Plan FY 04-FY08

**Barbara Lingberg**  
AIR 120

### FAA Aircraft Engineering Div.

*Barbara Lingberg is a Software Program Manager in the FAA's Aircraft Engineering Division and a program sponsor for FAA Software and Digital Systems Safety Program. Prior to her work in aircraft certification, she was software lead for the FAA's Wide Area Augmentation System. She holds a BS in mathematics and an MS in software systems engineering.*

Lingberg outlined the FAA's five-year plan for software. This year a great deal on Complex electronic hardware has been added. A snapshot of the FY03 tasks are the publication of:

AC 20-145 Guidance for Integrated Modular Avionics (IMA) that implements TSO-C153 Authorized Hardware Elements (published Feb. 03)

Order 8110.49 Software Approval Guidelines (Published June 03)

AC 20.RSC Reusable Software Components (Public Comments Obtained July 03)

AC 20.CEH Complex Electronic Hardware (Comments obtained Aug. 03)

In addition, the Certification Authorities Software Team hosted meeting number 37 and covered current topics on:

- Merging high and low-level requirements
- Data-base evaluation criteria
- Structural coverage of object code
- Reverse engineering
- Dealing with cache in certification projects

Future topics include:

- Dead vs. deactivated code
- CRCs
- Tracing structural coverage to requirements
- Compiler and math libraries
- Security assurance
- Probabilities and software testing
- Hardware vs. software vs. firmware
- Advanced verification methods for DO-254

These future issues are creating the need for a DO-178C. Current plans are to seek industry input now and launch the committee in Sept. 05.

Training programs in IMA/TSO, software job functions, real-time software development, and software for managers, CNS and

common civil/military software needs are also part of the FY03 task list.

Reports completed to date include a report on structural coverage of object code, (Nov. 02) and a study of COTS in real-time operating systems (RTOS) in aviation applications.

Highlights for FY04 are the formation of RTCA SC 200 collaborating with EUROCAE WG 60 to co-develop a standard for IMA. This work will extend out to FY06 and include an IMA platform Advisory Circular due June 06.

Also planned are handbooks, videos, and training courses for FAA software engineers and DER's on object-oriented applications and concerns for the aviation industry. Prototype training courses are expected by Oct. 05.

Plans are also under way for a Software & Digital Systems Safety (SDSS) program under the sponsorship of AIR 120. A small sampling of the topics to be covered include:

- Component integration
- LAN usage in aircraft
- OOT verification
- Accelerated Semiconductor device wear-out
- Ethernet as an aviation bus
- Case study on DO-254

Much of this research will be done in collaboration with NASA, various universities, Aerospace Vehicle Systems Institute (AVSI), Airworthiness Assurance Center of Excellence, Volpe National Transportation Systems Center, and the National Security Agency (NSA).

Finally, the long-overdue work on database certification issues is being addressed with the development of an AC next year and planned publication in Jan. 05.

## FAA Order 8110.49 Software Approval Guidelines

**John Lewis**  
Software Specialist  
FAA Aircraft Certification Service  
AIR 120

*John Lewis is a computer engineer and software specialist for AIR 120 in Washington, D.C. He has experience in developing FAA Notices, Orders, TSO's, and advisory circulars. He currently serves as secretary for RTCA SC 200 for IMA. John graduated from Virginia Tech with a BSEE and from Florida Tech with a Masters in Engineering Management and Business Administration.*

John reported on the consolidation of the 8110 series of FAA Notices into a single Mega-Order 8110.49 with 12 chapters. 8110.49 incorporates 11 existing Notices and was coordinated within the FAA, industry and the public:

Chapter	Topic
1	Introduction
2	Software Review Process
3	LOFI* in software Projects

(cont. on p. 8)

4	SW Conformity
5	Approval of FLS
6	PMA of FLS
7	Approval of UMS
8	Previously developed Software
9	Qualification of software tools
10	Software Changes in Legacy systems
11	SW Change Impact Analysis
12	Reused software life cycle data

\*Level of FAA Involvement

Most of the information in these chapters was preserved from the original release of the mega-order, with the exception of Chapter 4 on Software conformity and Chapter 6 on PMA via licensing agreements, which were significantly revised.

### Update of the Software Review Job Aid

**Leanna Rierson**

**National Resource Specialist**

*Leanna Rierson is the FAA's chief scientist and Technical Advisor for Aircraft Computer Software since 1999. She has over 15 years' experience in the computer/aviation industry. She graduated summa cum laude with a bachelor's degree in electrical engineering. She also holds a master's degree in software Engineering and is pursuing a doctorate.*

The Software Review Job Aid was first released in 1998 to provide a tool for standardized reviews by FAA engineers and designees and to improve the quality of the reviews. The purpose of the update is to address policy that has change and matured, correct errors, and implement lessons learned.

The tool has four parts:

Overview

Software Review Tasks

Stages of Involvement (SOI) Activities and Questions

Summary of Compliance/Findings

The Job Aid is built around the framework and requirements of DO-178B and the guidance of 8110.49. The stages of involvement are the classical R-D-C-I life cycle stages and the Aid provides guidance to the review of the artifacts of those stages.

Many of the changes were editorial in nature but some added new material to accommodate developments in OO, the real-time development course, and the new 8110.49 order.

### IMA Guidance Overview and SC-200 Status

**Leanna Rierson**

Integrated Modular Avionics (IMA) is defined as

*"...a shared set of flexible, reusable, and inter-operable hardware and software resources that create a platform that provides services, designed and verified to a defined set of safety and performance requirements, to host applications performing aircraft-related functions."*

IMA requirements are governed by TSO-C153 and AC-20-145. The TSO governs the hardware structure and requirements of the "brain-dead" black box. It allows manufacturers to stock and ship generic components and add functional software later. It allows the workload to be broken into small pieces and support electronic part marking and facilitates configuration

management. Each hardware element must have a data sheet to summarize its characteristics. The data sheet becomes part of the TSO authorization letter.

The Advisory Circular supplements TSO-C153 with 21 sections addressing safety assessment, configuration management, electronic ID, software, complex hardware, design guidance, environmental qualification, airworthiness, third-party issues, and many more. Software is addressed in Section 12 and points mainly to DO-178B for software assurance with references to the Field Loadable software chapters of 8110.49. It also addresses partitioning issues discussed in CAST papers and DO-248.

Later sections deal with roles and responsibilities of the C153 applicant, the functional TSO applicant, and the TC/STC applicant.

Bringing all this to a head will be the work of SC 200, chartered in March 2002. In August 2002 EUROCAE WG 60 joined SC200. The goal for the final guidance document is October 2004.

### Manufacturing/Production Software Activities

**Steve Sandmann**

**FAA**

*Steve Sandmann has 25 years of aviation experience within federal civil service, 23 in the DOD and two in the FAA. Before joining the FAA he spent a year at NASA, Kennedy Space Center, as a quality assurance lead for the solid rocket booster refurbishment. He is completing his bachelor's degree in quality management from the University of Phoenix.*

Sandmann presented the purpose of the Overarching Aviation Safety Inspector's Software (OASIS) Team. Chartered by the FAA Manufacturing Integration Management Team (MIMT) to develop a policy, guidance and training needs related to manufacturing and production software, the FAA is looking for better control of nondeliverable software. The current focus is on Computer-Aided Design, Manufacturing, Inspection and Test (CADMIT) software.

Toward that end, Sandmann serves as the FAA representative on the America's Aerospace Quality Group's Project 21, which is poised to release a standard for nondeliverable software. This standard fills the long standing void of controls for manufacturing software approval, configuration identification and control, release, archive, and retrieval. This standard is targeted for release late this year. It is harmonized with ISO 12207, 9000-3 SEI CMM, CMMI, and the FAA Advisory Circulars.

This standard will be a companion standard to AS9006, the aerospace quality system standard for deliverable software release in March by SAE. Both are framed around AS9100A, the aerospace quality standard that implements ISO 9001:2000.

The need for this standard is underscored by the prediction that 60% of aerospace product will be built offshore in the next six to eight years. The need for control of the ever-growing quantities of manufacturing software will be magnified accordingly.

### Software Service History

#### Research Results and Using the Handbook

**By Tom and Uma Ferrell**

**Ferrell and Associates, Consulting**

*Tom Ferrell is the co-founder of Ferrell and Associates Consulting Inc., a certification and software safety consultancy serving the safety and mission critical software industry. He holds a bachelor's degree in electrical engineering from Northern Illinois University, and a Masters degree in Information Technology Management from Rensselaer Polytechnic Institute. Uma Ferrell holds a Master's degree in*

*Electrical Engineering from Johns Hopkins University. She also holds a Master's degree in Solid State Physics and Bachelors degrees in Physics, Chemistry, and Mathematics from Bangalore University in India.*

Service history is one of the "alternate methods" discussed in DO-178B. Although allowable, service history has proved extremely problematic because of difficulty in proving the relevance of the environment, consistency of measurement, difficulties in defining and computing error rates, and effectiveness of problem solving.

Tom Ferrell reviewed the outline of DOT/FAA/AR-01/116, *Software Service History Handbook*. The handbook provides an overview of the 11 guidance statements for the use of product service history found in DO-178B. The handbook provides a series of worksheets and supporting information to facilitate the application of product service history as discussed in DO-178B. Over the last year, FAA Consulting conducted a follow on effort during which the handbook was used to evaluate the Level D software contained in the Wide Area Augmentation System (WAAS) for the purpose of reducing the long-term maintenance costs of this software. A 60-day study was used. This presentation is a report out of that effort. The Ferrells meticulously detail the results of their research reaching these conclusions and recommendations:

- Use of service history will not alleviate the need for continued documentation and development processes.
- The study's resultant failure rates cannot be used since 178B explicitly prohibits the use of software failure rate data in safety calculations.
- Robust problem reporting is required to keep the service history option open. Lack of sufficient granularity in problem reporting and instability of the system affect the ability to validly apply service history.

The Ferrells conclude that many more questions remain. For example,

- Why is the DO-178B service history discussion couched in terms of duration only without more consideration of coverage since this is emphasized in other areas of DO-178B?"
- How does one truly know to what extent the software was executed during the service history period - both a coverage and an environment issue?
- Would it be better to limit service history only to portions of software whose "inner workings" are not known and hence, there is no possible maintenance on that software?

### **Database Integrity (Navigation Databases)**

**Jeff Meyers**  
**FAA Airplane and Flight Crew Interface Branch**  
**Transport Airplane Directorate**  
**ANM-111**

*Jeff Meyers currently serves as navigation specialist on the Standards staff within the Transport Airplane Directorate. He has 19 years' experience in automatic flight controls and flight management systems development while working at McDonnell Douglas, Boeing, Learjet, Honeywell, and the FAA.*

Current Type Certifications do not address navigation databases. Navigation data consist of information on flight path, locations of waypoints, VOR (VHF omni-bearing range) Distance Measuring Equipment (DME), Standard Instrument Departures

(SIDS), Standard Terminal Arrival routes (STARs), and other information essential to the pilot and navigator.

Current operational rules address navigation databases but do not support the Aeronautical Data Chain concept. New operations are requiring database assurance, for example:

- P-RNAV requirements in JAA TGL-10
- GPS WAAS
- RNP RNAV

Some of the reasons for improving data integrity include:

- Source errors
- Incompatibility between data supplier coding and FMS decoding
- Data processing errors
- No clearly defined organizational responsibility
- Varying capabilities between avionics systems

The scope of impact is not trivial. There are 1000+ different design approvals. There are five FMS vendors, five GPS vendors, and roughly 10 products per supplier. There is only one U.S. domestic supplier of data, (Jeppesen), and two international suppliers (Lido, EAG). In addition there are hundreds of FAR Part 121 operators and thousands of FAR Part 91 operators. Since data are updated every 28 days, the opportunity for errors is immense.

Meyers described two different approval paths:

- Data verification
- Approved suppliers

The first uses a database-checking tool, the latter uses AC 90-DB to approve suppliers of data.

The tool uses a "gold standard database" and asks the question,

"Does the change have a significant adverse affect on the resulting flight path of the aircraft or information displayed to the crew?"

The advantage of this method is that it does not require approval of the supplier to DO-200A; however, the data verification can be difficult and expensive.

The other method (approval of suppliers) requires verification that the supplier and his data sources, configuration control, and quality assurance processes are working in accordance with RTCA/DO-200A. The advantage of the supplier approval method is that it provides more flexibility in obtaining approvals and accommodating changes. It is tailored to the most closely matched JAA database approval process. A disadvantage is that it requires training and coordination with the JAA.

The next step is to develop an integrated AC 90-DB that addresses both types of approval. This is currently being done through the Terminal Area Operations Aviation Rulemaking Committee (TAORAC). An FAA industry review of this draft is expected at the next TAORAC meeting planned for this month.

### **Software Conformity Inspection**

**Dennis Wallace**  
**ASW-170**

Chapter 4 of 8110.49 deals with software conformity. Earlier versions of the notice (FAA Notice 8110.86) were confusing. The current version helps clarify the distinction between software conformity product inspection and software conformity installation inspection.

Conformity, in its broadest sense, is compliance with the type design. Compliance (or conformance with the type design

*(cont. on p. 10)*

is achieved in two ways, both of which are required for total conformance:

Test Acceptance: Software Part Conformity Inspection (SPCI)

Installation Acceptance: Software Installation Conformity Inspection(SICI)

Part conformity, as with hardware, simply means the part conforms to its approved drawing. In the case of software, the software drawing is the Software Configuration I(SCI), variously called the version description drawing, version description document, etc. The SCI is the top assembly drawing for the software and must trace to all the life cycle artifacts, the requirements, the design, the source code, all the changes and problem reports, all the test artifacts, the support software and the build and load instructions. Para 8.3 of RTCA/DO-178B establishes part conformity requirements for software. Software Part Conformity Inspection is declared on an 8120-10 form.

Software Installation Conformity Inspection takes place after SPCI and assures that the correct (authorized) software is installed on the correct airplane. This is authorized in two ways. Either the software is embedded in a Line Replaceable Unit, for which a SPCI is completed; then the LRU is verified to be authorized by the airplane installation drawing; or the software may be directly loadable on the aircraft (sometimes known as Field Loadable Software (FLS)). This is declared on an 8110-1 form.

**Summary** :The foregoing is just a sampling of the papers presented that I was able to attend. Other papers were presented on:

Object Oriented Technology in Aviation  
RTCA/DO-254

Certification approaches using Ethernet-based aviation data buses.

Fault Tolerant IMA- the SPIDER project  
(Scalable Processor-Independent Design for Electromagnetic Resilience)

Previously Developed Software  
Software Changes in Legacy Systems  
Change Impact Analysis  
Commercial Derivative/ Military Aviation Software.

Unreachable code  
Software Tool qualification.  
Partitioning and Protection  
Thread Analysis

### About the author:

Mike Kress is an associate technical fellow for the Boeing Commercial Airplane Group in Seattle. He is a Senior member of ASQ and the chair of ASQ's Software Division. He is a Boeing Enterprise software process improvement facilitator and inter-divisional focal for software quality standards. He works with suppliers of aviation software to promote reliable processes for software development and maintenance. He is the author of "D1-9001, Advanced Quality System for Software Development and Maintenance", a supplier initiative dealing with an adaptation of the SEI-CMM for Boeing suppliers. Mr. Kress holds a bachelor's degree in electrical engineering as well as ASQ CQE, CSQE certifications and is a registered professional engineer. He has over 30 years' experience in military and commercial avion-

ics systems. He has served on numerous industry advisory and regulatory groups including AIA, AEA, RTCA, FAA SSAC, ARINC, and ISO. He is a member of the U.S. Technical Advisory Group to ISO/IEC SC7 TC176, and is the chair of AAQG Project 19, Deliverable Software Quality System Requirements.

---

## JOEL GLAZER—NEW ASQ FELLOW

---

The board of directors of ASQ, meeting in session in November 2003 in Milwaukee, has advanced **Joel Glazer** to the membership grade of Fellow.

From the Central Asian Republic of Kazakhstan, through post-WWII Europe's Displaced Persons camps and Israel, Glazer came to the United States at the age of 17. He earned a bachelor's degree in aerospace engineering from the University of Maryland, and received two master's degrees from Johns Hopkins University in management sciences and in computer sciences. His affiliation with ASQ began in 1987 when he was instrumental in establishing what became the ASQ Software Division, later joining ASQ itself in 1990. He assumed leadership roles as early as 1991 when he became a member of the Baltimore Section executive board, and has risen to participate in the Software Division's Council. So far, he has earned four certifications: Quality Audit, Quality Management, Software Quality Engineer, and Reliability Engineer. His broad experience includes working for Martin Marietta, Fairchild Hiller, Computer Sciences Corp., and E.G.G. on tasks ranging from aircraft design, rocket and space probes design, space travel, orbital dynamics, and earthquake predictions. In 1976 he joined Westinghouse - Baltimore Division, now Northrop Grumman ES, as a senior software engineer. Currently he is a Fellow Engineer in the Software Quality Engineering group. Since 1985 he has been at the core of the location's Software Quality group. As a software quality engineer, Glazer participated in establishing national and international software standards and represented the company on several national committees.

He has two married children, Hillel Glazer- Principal of Entinex, Inc. in Silver Spring MD, and Dr. Sharon Glazer— professor of Industrial Organizational Psychology at San Jose State, CA. He and his wife, Rachel, live in Maryland.

His Fellow citation, to be presented during the Annual Quality Congress in Toronto in May 2004, will state:

*"For outstanding contributions and support of ASQ, the Baltimore Section, and the Software Quality Division, with particular recognition of efforts to found and mentor the division, for continuing support as officer and advisor to Baltimore Section, and for expertise and willingness to train and mentor in the emerging field of Software Quality.*



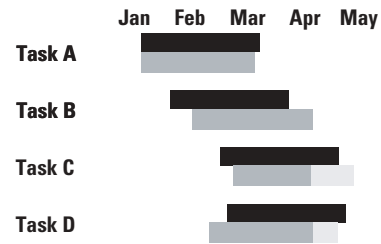
# SOFTWARE QUALITY ENGINEERING QUIZ

BY LINDA WESTFALL

Whether you are preparing for the Certified Software Quality Engineer (CSQE) examination or just testing out 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. \_\_\_ 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 the CSQE exam development process.*

1. Accepting an internal auditing assignment to audit a team's conformance to a procedure would be a potential conflict of interest if:
  - a. the procedure was written by another member of your SQA team.
  - b. you know the manager of the team being audited.
  - c. you had previously audited the same team against other procedures.
  - d. you are hoping to transfer to the team being audited.
2. In response to the results of an audit, a corrective action plan is required for:
  - a. each organizational area evaluated during the audit.
  - b. each finding documented in the audit report.
  - c. each observation documented in the audit report.
  - d. each nonconforming product identified during the audit.
3. A team of stakeholders including developers and end-users is brought together in a facilitated session to determine the requirements for a new release of a software product. This is an example of:
  - a. joint applications developm
  - b. quality functional deployment.
  - c. a focus group.
  - d. use case analysis.



**This Gantt chart is used in question 4**

4. Based on the Gantt chart shown above, which of the following tasks was completed behind schedule?
  - a. Task A
  - b. Task B
  - c. Task C
  - d. Task D
5. The mode of a data set is an indicator of:
  - a. the measurement scale used to collect that data set
  - b. the variation in that data set
  - c. the central tendency of that data set
  - d. the average of that data set
6. The valid inputs for variable X are character strings 1 and 32 characters in length. Which of the following sets of test case inputs should be selected to perform boundary testing on this input?
  - a. Null character string and character strings consisting of 1, 32, and 33 characters in length
  - b. Character strings consisting of 1, 2, 31, and 32 characters in length
  - c. Character strings consisting of 1 and 32 characters in length
  - d. Null character string and a 33 character strings
7. Which of the following tools would be considered part of the software configuration management toolset?
  - I. Virus detection and removal tool
  - II. Capture and playback tool
  - III. Change request reporting tool
  - IV. Code analysis tool
  - a. II only
  - b. I and III only
  - c. II and IV only
  - d. I, III, and IV only

Answers can be found on page 19

# STANDARDS CHAIR REPORT

BY SCOTT DUNCAN

Since there has been an unfortunate overlap in timing of IEEE and SC7 meetings and when newsletter issues have had to be prepared, I thought it would be good to simply present a summary of all the IEEE SESC and ISO/IEC JTC1/SC7 standards as a reference for folks. The next IEEE SESC meeting is in late February 2004 and the next US SC7 TAG meeting is in mid-April.

## IEEE SESC Standards Summary

Standard Number	Standard Name	Status/Notes
Adoption of ISO 15288	System Engineering	
ANSI/IEEE Std, 1008-1987(R1993) (App Dec. 11 '86, Reaff Dec. 2 '93) Reaffirmed Dec. 2002	An American National Standard - IEEE Standard for Software Unit Testing	
EIA/IEEE J-STD-016-1995 (Sept. 30)	Trial-Use Standard - Standard for Information Technology Software Life Cycle Processes Software Development - Acquirer-Supplier Agreement	This standard has been allowed to expire.
IEEE Std. 1012-1998 (Mar. 9)	IEEE Standard for Software Verification and Validation	Working group has held one meeting (spring 2003 in the Washington, DC, area).
IEEE Std. 1012a-1998 (Sept. 16)	Supplement to IEEE Standard for Software Verification and Validation: Content Map to IEEE/EIA 12207.1-1997	Next revision of 1012 should consider whether to incorporate this information in the Std.
IEEE Std. 1016-1998 (Sept. 23)	IEEE Recommended Practice for Software Design Descriptions	Draft 4.0 is projected for balloting in mid-January 2004.
IEEE Std. 1028-1997 (Mar. 4) Reaffirmed Sept. 2002	IEEE Standard for Software Reviews	
IEEE Std. 1044-1993 (Dec. 2) Reaffirmed Sept. 2002	IEEE Standard Classification for Software Anomalies	Chair needed for next revision effort.
IEEE Std. 1045-1992 (Sept. 17) Reaffirmed Dec. 2002	IEEE Standard for Software Productivity Metrics	
IEEE Std. 1058-1998 (Dec. 8)	IEEE Standard for Software Project Management Plans	Working Draft of revision exists.
IEEE Std. 1061-1998 (Dec. 8)	IEEE Standard for a Software Quality Metrics Methodology	Reaffirmation in progress.
IEEE Std. 1062, 1998 Edition (Dec. 2) Reaffirmed Sept 2002	IEEE Recommended Practice for Software Acquisition	WG for revision has not yet been formed.
IEEE Std. 1063-2001 (Dec. 5)	IEEE Standard for Software User Documentation	Action required in 2005.
IEEE Std. 1074-1997 (Dec. 9)	IEEE Standard for Developing Software Life Cycle Processes	Further information is available at the project Web site, <a href="http://www.p1074-workgroup.org/">http://www.p1074-workgroup.org/</a>
IEEE Std. 1175.1-2002 (Nov. 11)	IEEE Guide for CASE Tool Interconnections - Classification and Description	
IEEE Std. 1175-1991 (Dec. 5)	IEEE Standard Reference Model for Computing System Tool Interconnections	Withdrawn standard; no longer endorsed by IEEE.
IEEE Std. 1219-1998 (Jun. 25)	IEEE Standard for Software Maintenance	Will be combined with ISO/IEC 14764.
IEEE Std. 1220-1998 (Dec. 8)	IEEE Standard for the Application and Management of the Systems Engineering Process	Goal of revision is to harmonize with ISO/IEC 15288, CMMI, and SWEBOK.
IEEE Std. 1228-1994 (Mar. 17) Reaffirmed Dec. 2002	IEEE Standard for Software Safety Plans	
IEEE Std. 1233, 1998 Edition (Apr 17) Reaffirmed Sept. 2002	IEEE Guide for Developing System Requirements Specifications	Study group developing an outline for combining 1233 with 830.
IEEE Std. 1320.1-1998 (Jun. 25)	IEEE Standard for Functional Modeling Language—Syntax and Semantics for IDEF0	Will reaffirm.
IEEE Std. 1320.2-1998 (Jun. 25)		
IEEE Std. 1320.2a	IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X 97 (IDEF object )	Will reaffirm.
IEEE Std. 1362-1998 (Mar. 19)	IEEE Guide for Information Technology System Definition-Concept of Operations (ConOps) Document	Eventually harmonized with the new 1233/830.
IEEE Std. 14143.1-2000 (Jan. 30)	Implementation Note for IEEE Adoption of ISO/IEC 14143-1:1998 Information Technology—Software Measurement—Functional Size Measurement— Part 1: Definition of Concepts	Action required in 2004.

### IEEE SESC Standards Summary (cont.)

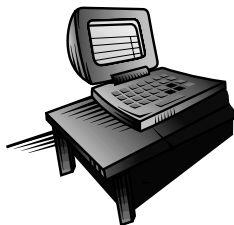
Standard Number	Standard Name	Status/Notes
IEEE Std. 14143.1-2000 (Jan. 30)	Implementation Note for IEEE Adoption of ISO/IEC 14143-1:1998 Information Technology—Software Measurement—Functional Size Measurement— Part 1: Definition of Concepts	Action required in 2004.
IEEE Std. 1420.1-1995 (Dec. 12) Reaffirmed June 2002	IEEE Standard for Information Technology—Software Reuse—Data Model for Reuse Library Interoperability: Basic Interoperability Data Model (BIDM)	
IEEE Std. 1420.1a-1996 (Dec. 10) Reaffirmed June 2002	Supplement to IEEE Standard for Information Technology—Software Reuse—Data Model for Reuse Library Interoperability: Asset Certification Framework	
IEEE Std. 1420.1b-1999 (Jun. 26) Reaffirmed June 2002	IEEE Trial-Use Supplement to IEEE Standard for Information Technology Software Reuse- Data Model for Reuse Library Interoperability: Intellectual property Rights Framework	
IEEE Std. 1448	Guide for Information Technology Software Life Cycle Processes	This project resulted in the IEEE adoption of ISO/IEC 12207:1995.
IEEE Std. 1462-1998 (Mar. 19)	IEEE Standard - Adoption of International Standard ISO/IEC 14102: 1995—Information Technology—Guideline for the evaluation and selection of CASE tools	Will readopt the revised version (or perhaps issue a corrigendum).
IEEE Std. 1465-1998 (Jun. 25)	IEEE Standard – Adoption of International Standard ISO/IEC 12119: 1994(E) – Information Technology – Software packages – Quality requirements and testing	Will readopt the revised version (or perhaps issue a corrigendum).
IEEE Std. 1471-2000 (Sept. 21)	IEEE Recommended Practice for Architectural Description of Software Intensive Systems	Action required in 2004.
IEEE Std. 1490-1998 (Jun. 25)	IEEE Guide - Adoption of PMI Standard – A Guide to the Project Management Body of Knowledge	IEEE adoption of the PMBOK -- 2000 edition. High priority.
IEEE Std. 1498-1995 (Sept. 21)	EIA/IEEE Interim Standard for Information Technology –Software Life Cycle Processes – Software Development: Acquirer-Supplier Agreement	Withdrawn. This became EIA/IEEE J-STD.-016-1995, which has been allowed to expire.
IEEE Std. 1517-1999 (Jun. 26) Reaffirmation process initiated.	IEEE Standard for Information Technology—Software Life Cycle Processes—Reuse Processes	Extension of 12207.0 to address software reuse.
IEEE Std. 1540-2001 (Mar. 17)	IEEE Standard for Software Life Cycle Processes – Risk Management	Will be combined with ISO/IEC 16085. See P16085.
IEEE Std. 2001-2002 (Jan. 21, 2003)	IEEE Recommended Practice for Internet Practices – Web Page Engineering –Intranet/Extranet Applications	Replaces 2001-1999.
IEEE Std. 610.12-1990 (Sept. 28) Reaffirmed Sept. 2002	IEEE Standard Glossary of Software Engineering Terminology	ISO/IEC SC7 will manage the next revision.
IEEE Std. 730-2002 (Sept)	IEEE Standard for Software Quality Assurance Plans	Next version will be a process std. A mid-2004 start is envisioned for this project.
IEEE Std. 828-1998 (Jun. 25)	IEEE Standard for Software Configuration Management Plans	New draft (including SWEBOK material) is essentially complete.
IEEE Std. 829-1998 (Sept. 16)	IEEE Standard for Software Test Documentation	WG is converting this document to a process standard.
IEEE Std. 830-1998 (Jun. 25)	IEEE Recommended Practice for Software Requirements Specifications	
IEEE Std. 982.1-1988 (Jun. 9)	IEEE Standard Dictionary of Measures to Produce Reliable Software	In December, RevCom will consider a recommendation to extend the standard until the PAR expires.
IEEE/EIA 12207.0-1996 (Mar.)	Industry Implementation of International Standard ISO/IEC 12207:1995 Standard for Information Technology –Software life cycle processes – Software Life Cycle Processes	Need to initiate a joint IEEE/EIA reaffirmation of the whole 12207 set.
IEEE/EIA 12207.1-1996 (April)	Industry Implementation of International Standard ISO/IEC 12207:1995 Standard for Information Technology –Software Life Cycle processes – Software Life Cycle Processes- life cycle data	We may also decide to reaffirm or issue a corrigendum.

## IEEE SESC Standards Summary (cont.)

Standard Number	Standard Name	Status/Notes
IEEE/EIA 12207.2-1997 (Apr 1998)	Industry Implementation of International Standard ISO/IEC 12207:1995 Standard for Information Technology –Software Life Cycle processes – Software Life Cycle Processes – Implementation considerations	We may want to ask for an extension until it's time for revision.
P1175.2	IEEE Guide for CASE Tool Interconnections – Characterization of Interconnections	Extension of the PAR until Dec. 31, 2005.
P1175.3	IEEE Guide for CASE Tool Interconnections – Reference Model for Specifying Software Behavior	Draft 4.02 is being revised.
P1175.4	IEEE Guide for CASE Tool Interconnections – Reference Model for Specifying System Behavior	WG is addressing several key technical issues. Hoping for ballot in 2004.
P1175.5	IEEE Guide for CASE Tool Interconnections – Syntax for Transferring Behavior Specifications	Work is ongoing. Balloting is not expected before 2004.
P14764	IEEE Standard for Software Maintenance	Joint revision of IEEE 1219; coordinated revision with ISO/IEC 14764.
P16085	IEEE Standard for Software Life Cycle Processes – Risk Management	Revision of IEEE 1540; coordinated revision with ISO/IEC 16085.
P1633 / AIAA R-013A	IEEE Recommended Practice on Software Reliability	This is a project of the Reliability Society.
P1644	IEEE Recommended Practice for Software Nomenclature – Software Naming Conventions for Application Software	An opportunity to collaborate with ISO TC 184/SC5/WG4, for coordination with ISO 16100.
P1648	IEEE Recommended Practice for Establishing and Managing Software Development Efforts Using Agile Methods	PAR in process.
Planned IEEE 14471		
Planned IEEE 15939		
Planned IEEE 9126.1		
Planned IEEE/ASQC 900003	Software and System Engineering – Guidelines for the Application of ISO 9001:2000 to Computer Software	IEEE/ASQC Adoption of ISO/IEC 90003.

## SQE & ISO CONSULTANT

### CSQE Online CLASS\*



**\$495** per person

- Course also available onsite
- ASQ section taught with a 95% pass rate
- Certified as a CQManager, CQA, CQE, CRE, CSQE, & RAB-LA
- Perform gap analysis & internal audits

\*Based on ASQ BOK

**ROBIN L. DUDASH**  
**INNOVATIVE QUALITY PRODUCTS & SYSTEMS, INC.**

phone/fax 724-789-7424  
iyps@aol.com • www.iyps.net

Thinking about taking the  
**ASQ CSQE Exam?**

The Westfall Team can  
help you get ready.



**Certified Software Quality  
Engineer Exam Refresher Course**  
Dallas, TX  
May 3-7, 2004



You will receive:

- In-depth review of all areas of the updated CSQE Body of Knowledge
- All presentation materials with annotated notes, glossary & index
- Sample questions with answers & explanations to practice taking the exam

For more information  
call 214-544-3694  
or visit

[www.WestfallTeam.com](http://www.WestfallTeam.com)

**The Westfall Team**

## SC7 Standards Summary

Std or TR	Work in Progress	Title	Comments re Status
IS 6592:2000		IT--Guidelines for the documentation of computer-based application systems	Published in 2000-03.
IS 9127: 1988, r94	CD 9127	Information processing systems — User documentation and cover information for consumer software packages	FCD ballot due 2003-10-09.
TR 9294: 1990	PDTR 9294	IT—Guidelines for the management of software documentation products	Will remain a TR.
	CD 15289	Guide for the application of ISO/IEC 12207 to the documentation process	Combined WD circulation, CD registration & CD ballot due 2003-09-30.
IS 15910: 1999		IT—Software user documentation process	Future revised NP likely.
	FCD 18019	Guidelines for design and preparation of SW user documentation	FDIS sent to ITTF for DIS ballot.
	15289	USNB contribution of IEEE/EIA 12207.1-1997 to WG2 for use in 15289	
IS 14102: 1995		IT—Guideline for evaluation and selection of CASE tools	Due for revision.
TR 14471: 1999		Guidelines for the adoption of CASE tools	
	FCD 15940	Software engineering environment services	FCD letter ballot passed.
	WD 18018	CM tool requirements	Combined WD, CDR, CD ballot authorized
IS 9126-1: 2001		Software engineering -- Product quality – Part 1: Quality model	
	TR 9126-2	Software engineering -- product quality – Part 2 : External metrics	Published.
	TR 9126-3	Software engineering -- product quality – Part 3 : Internal metrics	Sent to ITTF for publishing.
	TR 9126-4	product quality – Part 4 : Quality in use metrics	Published.
	9126-10	Software Engineering - Software quality – General overview, reference models and guide to software product quality requirements and evaluation (SQuaRE)	Re-numbered to ISO/IEC 25000.
	PNWI 9126-11	Software product quality – Part 11: Planning and management	
	PNWI 9126-20	Software product quality – Part 20: Quality model and guide	
	WD 9126-30	Software engineering – Software quality requirements and evaluation – Part 30: Quality metrics – Metrics reference model and guide	CD reg due 2002-04-13.
	NP TR 9126-31	Software product quality requirements and evaluation – Part 31: Quality metrics – Base metrics	NP approved.
	PNWI 9126-32	Software product quality requirements and evaluation - Part 32: Quality metrics – Internal metrics	
	PNWI 9126-33	Software product quality requirements and evaluation - Part 33: Quality metrics – External metrics	
	PNWI 9126-34	Software product quality requirements and evaluation - Part 34: Quality metrics – Quality in use metrics	
	PNWI 9126-35	Software product quality requirements and evaluation - Part 35: Quality metrics – Documentation of evaluation modules	
	WD/CD 25030 (formerly 9126-30)	Software engineering—Software quality requirements and evaluation (SQuaRE): Quality requirements	WD/CD Registration ballot due 11-14-02
	PNWI 9126-50	Software product quality requirements and evaluation – Part 50: Quality evaluation overview and guide	
	PNWI 9126-51	Software product quality requirements and evaluation - Part 51: Process for developers	
	PNWI 9126-52	Software product quality requirements and evaluation - Part 52: Process for acquirers	
	PNWI 9126-53	Software product quality requirements and evaluation - Part 53: Process for evaluators	

### SC7 Standards Summary (cont.)

Std or TR	Work in Progress	Title	Comments re Status
IS 12119: 1994	CD 12119	IT—Software packages quality requirements and testing	CD ballot passed 11-3-1.
		Software Engineering - Product evaluation - ...	
IS 14598-1: 1999		Part 1 : General overview	Published.
IS 14598-2: 2000		Part 2 : Planning and Management	Published.
IS 14598-3: 2000		Part 3 : Process for Developers	Published.
IS 14598-4: 1999		Part 4 : Process for Acquirers	Published.
IS 14598-5: 1998		Part 5 : Process for evaluators	Published.
IS 14598-6: 2001		Part 6 : Documentation of evaluation modules	Published.
IS 14756: 1999		Measurement and rating of performance of computer-based software systems	Published.
	CD2 25000 (formerly 9126-10)	Software and System Engineering: Software Product Quality Requirements and Evaluation (SQuaRE) - Guide to SQuaRE	CD2 ballot passed 11-2-3.
	CD 25020	SQuaRE: Quality Measurement - Measurement reference model and guide	CD ballot passed 13-2-2.
	WD/CD 25021	SQuaRE: Measurement primitives	WD/CD Registration ballot passed 11-2-3.
	CD 25030 (formerly 9126-30)	Software Engineering -- Software quality requirements and evaluation (SQuaRE): Quality requirements	CD ballot due 2003-06-03.
IS 12207: 1995		IT – Software life cycle processes	ISO/IEC 12207:1995/Amd. 1:2002 Draft corrigendum letter ballot due 2003-10-14.
	PNWI 12207	Revision of 12207	Study period authorized to harmonize 12207, 15288, 90003 and 15504.
	FDAM 12207/1	Amendment to 12207	
		Second amendment to 12207	
TR 15271: 1998		IT – Guide for ISO/IEC 12207 (Software Life Cycle Processes)	Published December 1998.
TR 14759: 1999		Software engineering--Mock up and prototype – A categorization of software mock up and prototype models and their use	Published.
IS 14764: 1999		IT—Software maintenance	Published. NP ballot on revision due 2003-09-09.
	IS 15288	System Engineering—System life cycle processes	Published November 2002
	NP 15288	Harmonization of 15288, 12207, 15504 including proposed revision of 15288	Study period authorized to harmonize of 12207, 15288, 90003 and 15504. NP letter ballot due 2003-10-01.
		Study Period for Systems Engineering Practices	Look at INCOSE work. Negotiate contributions from IEEE and EIA.
	DTR 19760	A Guide for the Application of ISO/IEC 15288 - System Life Cycle Processes	Sent to ITTF for TR processing.
	14399	Mapping of relevant software engineering standards--Standards relevant to ISO/IEC JTC1/SC7 software engineering	Cancelled after DTR. To be published as an SC7 standing document.
TR 12182: 1998		Categorization of software	
IS 15026: 1998		IT—System and software integrity levels	
	NP 15026	IT—System and software integrity levels	Revise to extend scope and link to safety and security standards. NP approved.
	DIS 16085	Software Life Cycle Processes--Risk Management (Fast-track of IEEE 1540)	SC7 authorized NP ballot. Passed DIS Ballot 15-0. Comments received from three nations.
	NP 16085 Revision	Software Life Cycle Processes--Risk Management	A revision project will deal with the comments received in DIS ballot.
	Study	Explore integrity concepts and applications and relationship with dependability	

### SC7 Standards Summary (cont.)

Std or TR	Work in Progress	Title	Comments re Status
	CD IEC 61720	Guide to techniques and tools for achieving confidence in software	Transfer from TC56 completed. Address as part of Part 5 of WG9 NP.
TR 15504-1: 1998		Software process assessment - Part 1 : Concepts and introductory guide	Published.
TR 15504-2: 1998		Software process assessment - Part 2 : A Reference Model for Processes and process capability	Published.
TR 15504-3: 1998		Software process assessment - Part 3 : Performing an assessment	Published.
TR 15504-4: 1998		Software process assessment - Part 4 : Guide to performing assessments	Published.
TR 15504-5: 1998		Software process assessment - Part 5 : An assessment model and indicator guidance	Published.
TR 15504-6: 1998		Software process assessment - Part 6 : Guide to competency of assessors	Published.
TR 15504-7: 1998		Software process assessment - Part 7 : Guide for use in process improvement	Published.
TR 15504-8: 1998		Software process assessment - Part 8 : Guide for use in determining supplier process capability	Published.
TR 15504-9: 1998		Software process assessment - Part 9 : Vocabulary	Published.
	FDAM 15504-2/1	Amendment to Part 2, for Acquisition Processes	
	WD/CD 15504-1 (IS)	Concepts and Vocabulary. (Revision of TR 15504 to a standard.)	CD ballot due 2003-07-03.
	FDIS 15504-2 (IS)	SWE - Process Assessment - Part 2: Performing an Assessment (Revision of TR 15504 to a standard)	
	FCD 15504-3 (IS)	Guidance to Performing an Assessment (Revision of TR 15504 to a standard)	Sent to ITTF for FDIS ballot.
	FCD 15504-4 (IS)	Guidance on use for Process Improvement and Process Capability Determination (Revision of TR 15504 to a standard)	FCD ballot passed 13-3-0.
	WD/CD 15504-5 (IS)	An Exemplar Assessment Model. (Revision of TR 15504 to a standard)	CD ballot due 2003-10-03.
IS 14143-1: 1998		Software measurement - Functional size measurement - Part 1 : Definition of concepts	NP for revision approved.
	FDIS 14143-2	Conformance assessment of software sizing methods	FDIS draft to be sent to CASCO as well as to JTC1 balloting.
	DTR 14143-3	Verification of functional size measurement method	Sent to ITTF for TR processing.
	TR 14143-4	Functional size measurement reference model	Submitted to ITTF for publication.
	DTR 14143-5	Determination of functional domains for use with functional size measurement	JTC1 ballot results on DTR were 14-2-2.
	NWI 14143-6	Information Technology - Software measurement - Functional Size Measurement - Part 6: Guide for use of ISO/IEC 14143 series and related international Standards	NWI proposal ballot due 2003-10-01.
	IS 15939	Software measurement process	Published.
	NP	Quality Framework--Guidance for the achievement of quality in software and systems	SSQF work moved to a subgroup of WG7.
	IS 90003	Guidelines for the application of ISO 9001:2000 for software	Submitted to ITTF for publication.
	DTR 19759	Software Engineering Body of Knowledge (SWEBOK)	DTR sent to JTC1 for balloting.
	WD 19770	Asset Management and SW License Management	Combined WD circulation, CD registration & CD ballot due 2003-10-03.
		SC7 Consolidated Vocabulary	NP ballot due 2003-10-01

[Those interested in standards work can contact Scott Duncan: [softqual@knology.net](mailto:softqual@knology.net), 706-649-2345 (weekdays), 706-565-9041 (evenings and weekends).]

## FROM THE REGIONS

### Region 4 Chris Fitzgibbon

This update to Region 4 (Canada) members provides information on conferences, local events, and opportunities to get actively involved with the Software Division.

#### Conferences

The dust has hardly settled from the 13th International Conference on Software Quality (ICSQ) in Dallas, and already attention has shifted to the next big event for ASQ members. On May 24 - 26, 2004, Toronto will host the ASQ's largest conference: the Annual Quality Congress. In addition to the convenience of having the conference in Canada, the 58th AQC offers an excellent set of speakers and topics that will inspire every quality practitioner in attendance. If you have never attended an AQC, this is an opportunity you won't want to miss! More information is available at the AQC Web site: <http://aqc.asq.org>.

#### Toronto and Southern Ontario

Also in Toronto, the **Toronto Association of Systems and Software Quality (TASSQ)** has dinner meetings during the last Tuesday of each month at the Sheraton Centre Toronto Hotel. Upcoming events are available at [www.tassq.org](http://www.tassq.org). Information on events by the **Toronto SPIN** and the **ASQ Toronto Section** is available from their Web sites: [www.torontospin.com](http://www.torontospin.com) and [www.asqtoronto.com](http://www.asqtoronto.com).

#### Ottawa and Montreal

The Ottawa Valley section's **Software Focus Group** has recently completed a successful nine-week study group to prepare participants for the CSQE exam. Its next CSQE study group is scheduled to run from April 1 to May 27, 2004. If you are interested in participating in a ASQ certification study group, contact me ([chris@orioncanada.com](mailto:chris@orioncanada.com)) and I will gladly forward additional information.

The Software Focus Group, in conjunction with the **Ottawa Software Quality Association (OSQA)**, also has a series of software quality presentations planned for this winter. Its most recent presentation was on "Process Improvement Experiences from CMM Level 3." More information on the OSQA and the ASQ Software Focus Group is available from [www.osqa.org](http://www.osqa.org) and [www.asqottawa.ca/software](http://www.asqottawa.ca/software).

### Calgary, Edmonton, and Vancouver

In September, a Vancouver organization of software quality practitioners held its inaugural meeting. **Software Quality Assurance Vancouver User Group (VanQ)**, which includes students in the software quality assurance course at Kwantlen University College (see *ASQ Software Quality Professional*, June 2003), holds regular meetings at the Burnaby campus of the British Columbia Institute of Technology. Recent topics have included: "Model-based Test Generation," "Automated Build and Testing With .NET," and "Strategies for Prioritizing Test Activities." Additional information about this new and exciting software QA peer group may be obtained from their Web site: [www.vanq.org](http://www.vanq.org).

A similar organization in Calgary, the IEEE/ASQ Discussion Group for Software Quality, has been successful for several years. Its upcoming presentation topics include "Putting Science Behind Software Estimating," "The Future of the Industry," and "Software Testing." The group meets every two weeks at the Calgary campus of the DeVry Institute from September through May. All sessions are free and advance registration is not required. Their Web site is [www.software-quality.ab.ca](http://www.software-quality.ab.ca).

If you have information that you would like to share with fellow ASQ Software Division members, or you would like to get involved with the Division, contact me at [chris@orioncanada.com](mailto:chris@orioncanada.com) or at 613-563-9000. It would be a pleasure to hear from you.

### Region 5 Joel Glazer

Planning for April Software Testing and Software Auditing Dinner Meeting for the Baltimore section.

### Region 6 Tom Gilchrist

The Pacific Northwest Quality Conference organization sponsored its annual quality conference October 13-15, 2003, at the Oregon Convention Center in Portland. Keynote speakers this year were Cem Kaner ("How Many Light Bulbs Does It Take to Change a Tester?") and Brian Marick ("Agile Testing: A Year in Review"). There were technical papers and other invited speakers as well as workshops and exhibits. For more information, visit <http://www.pnsgc.org>.

On the third Thursday of every month (except December), SASQAG holds public meetings in the Seattle area at

Attachmate in Factoria. SASQAG also supports certification and study groups. If you are in the area and want to attend, please look at [www.sasqag.org](http://www.sasqag.org) for upcoming events, directions, and meeting time.

If you have information on local software quality and testing events in your area of Region 6, please send them to me for our events calendar. I am looking for more information about activities and events in California. Visit <http://www.tomgtomg.com/asq6> for information on events around Region 6.

### Region 10 Nancy Poma

If you are interested in speaking or planning the 5th Annual Quality Conference in Michigan for next October, please contact me at [nmpoma@comcast.net](mailto:nmpoma@comcast.net). Remember that you earn credits toward your ASQ recertification for these activities as well as a complimentary registration for the conference.

The next GL-SPIN meeting is January 8, 2004, at U of M Dearborn, the following meeting is February 12, 2004, at Oakland University, and you can get more information on both of these events at [www.gl-spin.org](http://www.gl-spin.org). Again, these meetings are a good way to earn recertification.

Finally, the next CSQE Refresher offered by the Greater Detroit Section will start on March 13, 2004, and you can get more information at [www.asqdetroit.org](http://www.asqdetroit.org). If anyone has other information that you want to distribute to software professionals in ASQ Region 10, please contact me at [nmpoma@comcast.net](mailto:nmpoma@comcast.net). I hope that 2004 brings you the best both personally and professionally.

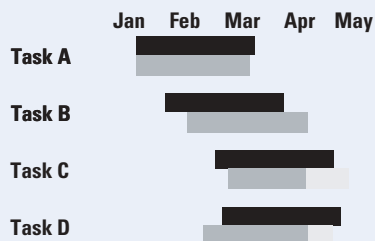
### Region 13 Granville Jones

- Continued helping with the 2004 Rocky Mt. Quality Conference April 26-27, 2004.
- Participated in the Nov .11, 2003, RC's teleconference.
- Continued serving as program chair for Denver Section 1300. Setting up a joint meeting with the Denver Chapter of the Institute of Industrial Engineers.
- Wrote and published an article for the Mile High Chapter of the Project Management Institute on customer satisfaction.

# ANSWERS TO THE SOFTWARE QUALITY ENGINEERING QUIZ

BY LINDA WESTFALL

- 1. Answer D is correct.** According to the ASQ Code of Ethics, a potential conflict of interest exists when “any business connections, interest, or affiliations that might influence my judgment or impair the equitable character of my service” exists. The desire to transfer to or be hired by the team being audited would be an example of a potential conflict of interest. Simply knowing the manager of the team being audited, previously auditing that team, or the fact that the procedure was written by another member of your SQA staff typically does not result in a conflict of interest. In internal auditing, these situations occur frequently. **CSQE Body of Knowledge Area: I.D.1**
- 2. Answer B is correct.** The corrective action plans are created by the auditee’s organization to address the root causes of the nonconformances and noncompliances documented as findings in the audit report. While some findings may be the responsibility of specific organizational areas that were evaluated during an audit, other finding may require cross-functional corrective action. Other organizational areas may have been found to be in compliance and therefore require no corrective action. Observations may be either positive or negative. While, observations do not specifically require corrective action as part of the audit process, it may be a good idea to evaluate the future impact of negative observations and take corrective actions as necessary. If only the specific nonconforming products are corrected, only the symptom may be addressed and not the root cause. Also since auditing is a sampling process, other nonconforming products may exist that were not examined during the audit. **CSQE Body of Knowledge Area: II.C.3**
- 3. Answer A is correct.** The purpose of joint applications development (JAD) is to bring together developers and users to jointly define an application or system. A JAD session is a facilitated workshop that streamlines communications by bringing together a cross-functional group of stakeholders to identify and resolve issues. **CSQE Body of Knowledge Area: III.C.2**



**4. Answer B is correct.** In this tracking Gantt chart, the black bars represent the original baselined schedule, the dark grey bars represent the actual status to date and the pale grey bars reflect the new projected schedule. Task B was started behind schedule and was completed behind schedule. Task A started on time and was completed slightly ahead of schedule. Task C started behind schedule, is still in progress, and is projected to complete behind schedule. Task D started ahead of schedule, is still in progress, and is projected to complete ahead of schedule. **CSQE Body of Knowledge Area: IV.A.3**

- 5. Answer C is correct.** The mode of a data set is the value that occurs most often in that data set. The mode is used to indicate the central tendency of data sets with severely skewed distributions, for irregular situations (e.g., where two peaks are found in the distribution of the data set), or for eliminating the effects of extreme values. For example, in the data set {1, 2, 2, 5, 6, 6, 6, 7, 8, 9, 12, 12, 1534} six is the mode. **CSQE Body of Knowledge Area: V.A.2**
- 6. Answer A is correct.** Boundary value analysis explores the values around the boundaries of the equivalence classes. This is done by testing:
  - the minimum value (character string of one character)
  - the maximum value (character string of 32 characters)
  - the value just below the minimum (null character string)
  - the value just above the maximum (character string of 33 characters).**CSQE Body of Knowledge Area: VI.C.4**
- 7. Answer B is correct.** A virus detection and removal tool would be part of the configuration management toolset because they ensure the integrity of the software build and help guarantee that only authorized changes are made to the configuration items. A change request reporting tool supports the change control function and would therefore also be considered part of the SCM toolset. A capture and playback tool would be used for test automation and is a testing tool. A code analysis tool would also be part of either the software engineering/development toolset or part of the verification and validation toolset depending on the type of analysis that the tool performed. **CSQE Body of Knowledge Area: VII.A.3**

---

# UTILIZATION OF DEFECT DENSITY METRIC FOR SPC ANALYSIS

---

K.U. SARGUT  
UNIVERSITY OF FLORIDA • GAINESVILLE, FL, USA

O. DEMIRÖRS  
MIDDLE EAST TECHNICAL UNIVERSITY • ANKARA, TURKEY

---

## Abstract

*In this paper, we describe the difficulties and suggestions in application of statistical process control to a CMM Level 3 organization using defect density metric. The paper discusses the defect density metric and demonstrates that the metric requires a precise definition of defect as well as product size for different phases of software development to be used for statistical process control. We suggest using XmR charts for tracking defect density instead of the popular u-chart, which depends on the assumption that the data has Poisson distribution. We also summarize the related results of a case study we have performed in a CMM Level 3 software organization to explore whether SPC can produce beneficial results for a software company.*

## Introduction

Since the industrial revolution, SPC (statistical process control) has been widely used in manufacturing industries in order to control and improve processes (Sutherland, Devor, & Chang, 1992). While benefits of SPC are well founded for manufacturing companies, there have been many debates (Card, 1994; Kan, 1995; Lantzy, 1992) about its application in software industry. It is frequently argued that the inherent characteristics of software cause difficulties in applying SPC techniques. As a result, SPC had not been widely used in the software industry until the 1990s.

The interest to apply SPC techniques in the software industry has been growing during the last decade as the number of organizations that advance in maturity levels of process improvement models such as Capability Maturity Model (CMM) (Paul, Weber, Garcia, Chrissis, & Bush, 1993), Capability Maturity Model Integration (CMMI) (CMMI Product Team, 2001), and SPICE (*Software Process Assessment*, 1998) increases. These models implicitly direct software companies to implement SPC as a crucial step for achieving higher process maturity levels. They suggest control charts for both project level process control and organizational level process improvement purposes. Many researchers contribute to this trend by providing approaches to utilize SPC techniques for software industry (Burr, & Owen, 1996; Fenton, & Neil, 1999; Florac, & Carleton 1999; Florac, Park, & Carleton, 1997). Moreover, most of the examples exhibited in the studies refer to defect and inspection data ((Burr, & Owen, 1996; Fenton, & Neil, 1999; Florac, & Carleton 1999; Florac, & Carleton, 1999 Symposium; Florac, & Carleton, 2000; Humphrey, 1989; Radice, 1998; Weller, 2000). These studies however, focus on the potential benefits of SPC implementation rather than providing a satisfactory guideline for software firms to implement SPC techniques with convincing practical evidence. We specifically lack knowledge on the applicability of different metrics, the means of reliable data collection mechanisms, meaningful analysis approaches, and practical evidence.

The need for such knowledge in the industry encouraged us to perform a study to show whether SPC techniques can be uti-

lized effectively in software development life cycle processes. We started with the idea that it is possible to use control charts for software processes with specified metrics provided that the processes are stable enough. We investigated difficulties and approaches to use SPC effectively by using the metric data that organizations frequently collect. Then we performed a case study in a CMM Level 3 software organization, which has 45 engineers and is experiencing a period of improvement to achieve CMM Level 4. During the case, we worked on the existing metric data related to established company-specific procedures and translated the metric data to a form that is appropriate for a comparison among different projects and products. As control chart is one of the most sophisticated data analysis tools within SPC, we demonstrated practical evidence on the utilization of SPC via control charts.

In this paper we present our observations and suggestions on the usability of defect density data for statistical process control for a CMM Level 3 organization. The paper can be analyzed in three sections. In the first section, we outline some major problems of measuring defect density metric, provide possible solutions, and describe the means of drawing control charts. In the second section, we demonstrate our experiences about defect density metric as part of the case study, and finally we summarize our findings.

## Defect Density: Problems and Solutions

The number of defects in a work product is an important measure as it gives us an intuition about how much the customer will be satisfied (from post-release defects), how much rework will be performed, how efficient our inspection processes work, which processes need improvement, which components of the system are error-prone. Therefore, defect counts provide evidence not only on the quality of the product, but also on the quality of the related processes.

In order to provide a basis for comparison between different software products, defect data are normalized by the addition of size dimension to the analysis. The resulting metric is defect density, which is defined as:

$$\text{defect density} = \text{number of defects} / \text{product size}$$

The analysis and interpretation of defect density metric lies on the assumption that, on average, we have a certain expectancy of defect count for unit size of a software artifact. As we desire to find most of the existing defects in a successful inspection, this measure is also an indicator of the effectiveness of the related inspection process.

Before starting to measure defect density, it is important to establish necessary background in order to provide consistency among measurements. First of all, the meaning of a defect should be defined precisely. Actually, it is common to come up against different definitions of this term in various resources. As being an international guide for software organizations, IEEE Std 982.2-1988 software standard (IEEE Guide, 1998) defines defect as a product anomaly: (1) omissions and imperfections found

during early life cycle phases, and (2) faults contained in software sufficiently mature for test or operation. For ease of understanding, we will refer to any software or documentation anomaly as a defect in our analysis.

Although a defect represents a software anomaly, each defect has distinctive properties. Not all defects have the same priority, impact, urgency, origination point, and cause. For this reason, it is essential to categorize defect data into specific groups up to a certain level. However, each software organization should individually determine the level of detail and the categorization of the groups after working on the historical data, determining specific needs and analyzing related processes. IEEE gives a practical framework for the classification of defect data (*IEEE Standard*, 1993; *IEEE Guide*, 1995).

In some cases, an anomaly about a functional software requirement might be regarded as a defect, although it may constitute more than one defect. For instance, a nonconformity detected by the customer during a qualification test may be recorded as “a new customer cannot be added by using addition function.” From the customer’s perspective, the defect is the inability to save a new customer record. When the problem is investigated, it may be realized that one of the fields (e.g. customer name) has a database connection problem, another field (e.g., customer NO) has a logic error while the confirmation button has a totally distinct problem. In order to avoid such misconceptions, defects having different origins, causes, and priorities should be recorded separately.

If we focus on the size of work products, we see that various software products have different size measures. For instance, it can be lines of code for software, number of pages or number of words for a document, function points for the whole project. Although defect density metric neutralizes the effect of size on defect count, there are still other parameters that should be considered in order to provide a firm basis for data interpretation.

Source lines of code is one of the major measures for counting the size of code. However, this is not as simple as its name implies. Source code is composed of different sections like executables, data declarations, and comment lines. Moreover, software tools provide enhanced capabilities for code generation, user interface development, and compilation. Without writing a single line of code, it is made possible to create a user interface by dragging and dropping components. However, all these complexities make it difficult to measure size (Fenton, & Neil, 1999). In order to obtain a concrete measure for quantifying software size, different SLOC sections should be measured separately.

Another difficulty during the interpretation of SLOC data arises from the existence of different programming languages. The number of code lines required to perform a function changes from one language to the other. If there are enough data, separate analyses may be performed for the measures in different programming languages. If the data are not enough, conversion tables may be used to convert the number of lines in one language to the Assembly equivalent. However, considering the defect density metric, it is difficult to decide whether to use converted lines or not. If a coding defect is due to a logical error, it would be reasonable to convert all source lines to assembly language. However, if the defect is due to a syntax error, the total lines of code would be more meaningful regardless of the language used. Obviously, such an analysis necessitates a detailed categorization of defect data.

Complexity is also a critical issue in SLOC evaluation. It is most probable that a complex code component will have more defects

than a simple one. Therefore, it is usually not reasonable to treat two source code statements if their complexities are not similar. In order to refine the analysis, it is also possible to include the complexity dimension by multiplying SLOC count with a complexity adjustment factor (such as cyclomatic complexity).

Integrated development allows software engineers to share their code with others and to utilize reusable code components. The important thing in measurement of shared code is to count it once as part of the component in which it is created. Likewise, code components include reused sections which are ideally believed to be error-free as they are inspected before. However, practically any code component has potential for containing defects, being less in reused source statements. For this reason, the extent and quality of reused code should be recognized so that different parts of code can be treated separately.

An alternative to physical SLOC measure is logical source lines of code, which can be more representative for the complexity issue. However, logical SLOC measure causes further problems regarding defect counts since the defect may be due to syntax errors. Jones (2000) outlines the main weaknesses and strengths of physical and logical lines of code measures. Although he reveals that logical statements are more rational for productivity and product quality measurements, he regards both of these measures as misleading. He finally suggests function point metric for productivity and product quality measurements.

When we consider defect density metric for document type products, we have to propose the means of measuring document size. The most frequently used document size measures are the number of pages and the number of words. Although both measures are easy to collect, there are important concerns that we should consider before forming a relationship between the number of defects and the document size. First of all, the physical properties such as font size, font type, page style, and margins are influential factors for the number of pages. Therefore, their consistency should be provided for meaningful comparison among different documents. Secondly, a document is made up of different components like words, figures, tables and graphs with different complexities, sizes and defect containment potentials. In order to avoid related variations, it is possible to compute a weighted size measure by giving different weights to different components. Another difficulty arises because of nonfunctional parts such as empty pages, cover page and pages including table of contents, index, signatures, and so on. These sections are not apt to have defects and should be disregarded during size calculation. Similarly, a document includes reused sections that are extracted from other documents. The reused sections should also be treated differently either by ignoring during size measurement or by counting a percentage of the reused pages/words depending on the extent of reuse.

Other external parameters beyond size and defect definition might also influence defect density measure. SPC requires rational sampling of data. That is, even after stabilizing the processes, it will be possible to apply SPC techniques only after parameters influencing metrics are appropriately defined and data are correctly categorized. Another difficulty in utilizing control charts with the metrics collected is related to the distribution of data. In most of the resources (Burr, & Owen, 1996; Fenton, & Neil, 1999; Radice, 1998; Weller, 2000), u-chart is suggested for tracking defect density data. However, u-chart depends on the assumption that the defect data follow a Poisson distribution.

(cont. on p. 22)

# SPC ANALYSIS

## CONTINUED

Therefore, it is very important to check for the validity of this assumption by analyzing metric data. On the other hand, the width of control limits for a u-chart is inversely proportional to the square root of sample size (product size). Therefore, having 10 defects in a sample of size 20 is not equivalent to having 100 defects in a sample of size 200, which is not reasonable for code defects. Although the chart may be useful if SLOC is counted for software units with similar sizes, it is usually more appropriate to use XmR charts instead. In the next section, we will demonstrate our case study via X (Individuals) charts for defect density data.

### Utilizing Defect Density for SPC

Initially, defect density metric is analyzed while performing research studies on the usability of SPC techniques. In the company, the data of all defects found during a review, test, or audit have been collected and tracked through Problem Reports (for code defects) and Document Change Requests (for document defects) since the foundation of the company in 1998. Although the trouble reports evolved within time, the basic defect information such as the subject work product, related project phase, defect priority, and initiation and closure dates are recorded for all the projects.

Each defect on a trouble report is given a priority, which is classified as low, medium, high, very high, and other. However, after collecting the data, we realized that there was not enough data from each priority category for a successful SPC analysis. Thus we combined five priority categories within three groups:

- Group 1: Combining high and very high
- Group 2: For medium
- Group 3: Combining low and other

While making this categorization, our assumption was that it was meaningful to pay similar attention to the defects in priorities low and other.

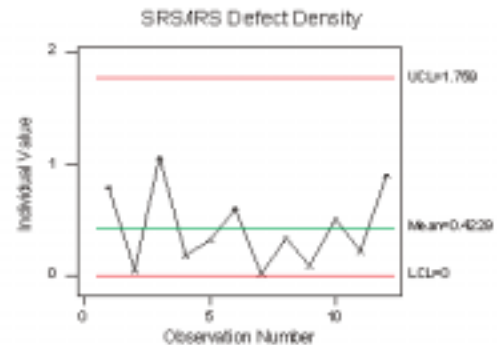
The code size is collected for each CSCI in terms of Source Lines of Code, excluding comment lines and blanks. As data are not available in software unit level, small number of projects did not allow us to perform SPC analysis to code defects. As a way of increasing the number of data samples, a weighted measure is calculated by multiplying defect densities with different weights depending on their priorities. Nevertheless, the distribution of data was too variable that we could not find a reasonable basis for our idea.

Considering process control purposes, we decided to restrict our analysis to requirements and design documents and defined size measures as the following.

1. Requirements documents (SRS and IRS): The number of requirements is used to compute size.
2. Design Documents (SDD and IDD): The number of pages is used to compute size.

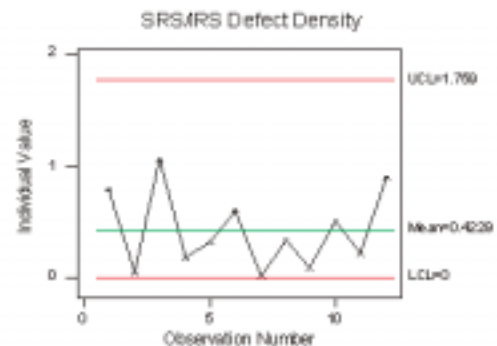
We computed the cumulative number of defects for each document. As the size of a document remains almost the same throughout the project, the defect density value gradually increases as more defects are detected. For this reason, we decided to make separate analyses by comparing the defect density values at the end of each project phase for two document groups. However, division of data among different priorities, project phases, and document groups highly reduced the affectivity of SPC analysis as the number of samples became insufficient. Therefore, we restricted our analysis for design and maintenance phases. As maintenance still continues for some of the projects, the data may need some adjustment at the end of maintenance phases.

The document size is gathered for each version and the size of the last version that is already released at the end of the project phase is used for defect density measurement. Afterward, Individuals charts are drawn for each 'project phase,' 'priority,' 'document type' combination. The Individuals chart for low priority defect density data at the end of the maintenance phase for requirements documents is shown in the following figure.



In the graph, each observation corresponds to a project or CSCI for which an SRS or IRS exists. Florac & Carleton ("Measuring the Software Process", 1999) state that it is desirable to have at least 40-45 individual values in order to draw a reliable Individuals chart. Although we only have 12 data points, we regard this as a good starting point for future applications of control charts.

The observations are in the order of the document preparation dates from past to future. We see that the process is under control and all the variation comes from inherent process characteristics. As a complement to the previous chart, we also drew Individuals chart for priority 3 design documents at the end of the maintenance phase. In this chart, observation 1 represents IDD document for Project 7. This extreme point shows us an abnormality in Project 7 because of some critical change in the interface design. Despite the scarcity of data points, Individuals chart acted as a means of comparing process performances statistically. Actually, detecting such a deficiency during the maintenance phase may not be beneficial to take any corrective action in order to solve process problems for the related project. However, the reasons for these nonconformities may be investigated and the results may be used for improvement actions.



We prepared similar charts for design and maintenance phases with different priorities. Unfortunately, however, we could not obtain high effectiveness. Some of the charts were useless since there were very few data, while some others showed too much variability leading to the control limits becoming too wide. Nevertheless, we obtained some evidence that process performance may be visualized by applying SPC techniques to defect density data provided that the data are sufficient and detailed enough.

### Summary and Conclusions

The application of SPC to software processes has been a challenging issue for software engineers and researchers. Although SPC is suggested for providing process control and achieving higher process maturity levels, there are very few resources that describe success stories, implementation details, and implemented guidelines for applying SPC to specific metrics.

In this paper, we presented an analysis of defect density metric. Based on the observations presented, we conclude that it is necessary to make a precise definition of defect and categorize defect data so that the data become meaningful for SPC analysis. Moreover, the size measure should be distinct and well-defined for different work products. While computing size, it is also important to obtain separate measures for different sections of the work products. We also showed that Individuals chart is more appropriate than a u-chart for analyzing defect density data.

In order to investigate the application of SPC to defect density data in a software production environment, we performed a case study in a CMM Level 3 organization. The case results revealed that it is difficult to find appropriate data to enable meaningful SPC analysis. Sometimes the data become insufficient to construct control limits, and sometimes the variability turns out to be so high as a result of an unstable process. Nevertheless, we could still capture an outlier by using control charts to defect density metric. This is encouraging evidence for the power of SPC to give us a view of a software process. Although the interpretation of the results is not as precise as it is in manufacturing, beneficial information can be gathered for process control and improvement purposes. The importance of obtaining an adequate amount of data also revealed that software organizations having a high number of projects can obtain better results from SPC implementation.

This study is expected to be a step to understand practical problems of statistical process control in the software industry. We identify a number of problems that should be addressed as separate research studies. For instance, the effect of different parameters (like programming language, defect priority, SLOC definition, etc., in defect density) on the results of SPC analysis may be investigated by a stepwise study with real metric data. Similarly, an interorganizational study might be performed to evaluate the effectiveness of the same SPC techniques in different environments.

### References

- Burr, A., & M. Owen M. (1996). *Statistical Methods for Software Quality*. Thomson Publishing Company. ISBN 1-85032-171-X.
- Card, D. (1994). Statistical Process Control for Software?. *IEEE Software*, May 1994, 95-97.
- CMMI Product Team. (2001). *CMMI<sup>SM</sup> for Systems Engineering, Software Engineering, and Integrated Product and Process Development (CMMI-SE/SW/IPPD, V1.1), Continuous Representation*. Carnegie Mellon University, December 2001.
- Fenton, N.E., & M. Neil (1999). Software Metrics: successes, failures and new directions". *The Journal of Systems and Software*, 47, 149-157.
- Florac A.W., E.R. Park, & A.D. Carleton (1997). *Practical Software Measurement: Measuring for Process Management and Improvement (CMU/SEI-97-HB-003)*. Software Engineering Institute, Carnegie Mellon University.
- Florac, A.W., & A.D. Carleton (2000). Statistical Process Control: Analyzing a Space Shuttle Onboard Software Process. *IEEE Software*, July/August 2000, 97-106.
- Florac, A.W., & A.D. Carleton (1999). *Statistically Controlling the Software Process (The 99 SEI Software Engineering Symposium)*. Software Engineering Institute, Carnegie Mellon University.
- Florac, A.W., & A.D. Carleton (1999). *Measuring the Software Process: Statistical Process Control for Software Process Improvement*. Pearson Education. ISBN 0-201-60444-2.
- Humphrey, W. (1989). *Managing the Software Process*. Reading, Mass.: Addison-Wesley Publishing Company. ISBN 0-201-18095-2.
- IEEE Guide to Classification for Software Anomalies, Std. 1044.1-1995.
- IEEE Guide to Use of IEEE Standard Dictionary of Measures to Produce Reliable Software, Std. 982.2-1998.
- IEEE Standard Classification for Software Anomalies, Std. 1044-1993.
- Information Technology - Software Process Assessment - Part 4: Guide to Performing Assessments (ISO/IEC 15504-4:1998(E)).
- Jones, C. (2000). *Software Assessments, Benchmarks, and Best Practices*. Addison-Wesley Longman Inc. ISBN 0-201-48542-7.
- Kan, S. H. (1995). *Metrics and Models in Software Quality Engineering*. Addison-Wesley Publishing Company. ISBN 0-201-63339-6.
- Lantzy, M.A. (1992). Application of Statistical Process Control to Software Processes. *Proceedings of the Ninth Washington Ada Symposium on Empowering Software Users and Developers (WADAS '92)*, 113-123.
- Paul, M.C., C.V. Weber, S.M. Garcia, M.B. Chrissis, M. Bush (1993). *Key Practices of the Capability Maturity Model, Version 1.1*. Software Engineering Institute, Carnegie Mellon University.
- Radice, R. (1998). Statistical Process Control for Software Projects. *10th Software Engineering Process Group Conference*. Chicago, Illinois.
- Sutherland, J., R. Devor, T. Chang (1992). *Statistical Quality Design and Control*. Prentice Hall Publishing Company ISBN: 002329180X.
- Weller, E., (2000). Practical Applications of Statistical Process Control. *IEEE Software*, May/June 2000, 48-55.

## OFFICERS

**Michael P. Kress**, Chair  
The Boeing Company  
425-717-7038  
michael.p.kress@boeing.com

**Doug Hamilton**, Chair-Elect,  
Strategic Planning Chair,  
Certification Chair  
Accenture  
312-693-0308  
douglas.b.hamilton@accenture.com

**Linda Westfall**, Nominating Chair,  
13ICSQ Chair  
The Westfall Team  
972-867-1172  
LWestfall@WestfallTeam.com

**Eva Freund**, Treasurer  
The IV&V Group  
703-573-7466 (voicemail)  
efreund@erols.com

**Terry Deupree**, Secretary  
JPMorgan Chase  
469-477-0362  
tdeupree@attbi.com

**Theresa Hunt**, Vice Chair Programs  
The Westfall Team  
Theresa\_Hunt@WestfallTeam.com

**Hank Sobah**, Vice Chair Member  
Services  
Innovative Systems, Inc.  
412-937-7687  
hsobah@innovativesystems.com

## CHAIRS & OTHER CONTACTS

**Sue Carroll**, Examining and  
Awards Chair  
SAS  
919-677-8000, ext. 17032  
Sue\_Carroll@Bellsouth.net

**Tom F. Griffin**, Publications Chair  
Auburn University—Montgomery  
334-244-3304  
tgriffin@mail.aum.edu

**Robin Dudash**, Education Chair  
iqps@aol.com

**Greg Simkins**, Internet Chair  
412-341-7926  
gregsim@telerama.com

**Jayesh G. Dalal**, Bylaws Chair  
732-591-0146  
jdalal@att.net

**Larry F. Jones**, 12 ICSQ Chair  
LFJ Group Inc.  
613-299-2770  
joneslf@magma.ca

**Taz Daughtrey**, Liaison Chair,  
Journal Editor  
804-237-2723  
sqpeditor@aol.com

**Patricia McQuaid**, World Congress  
California Polytechnic State  
University  
805-756-5381  
pmcquaid@calpoly.edu

**Carol Dekkers**, AQC Track Chair  
Quality Plus Technologies, Inc.  
727-393-6048  
dekkers@qualityplustech.com

**Rufus Turpin**, Marketing Chair  
Carpe Diem Infomatics, Inc.  
613-715-9146  
rufus@carpedieminfo.ca

**Scott Duncan**, Standards Chair  
706-649-2345  
softqual@knology.net  
**David Walker**, Regional Councilor  
Coordinator  
Pfizer Corporation  
269-833-7919  
david.w.walker@pfizer.com

## REGIONAL COUNCILORS

**Region 1**—Eric Patel  
RapidSQA  
877-749-4586  
epatel@rapidsqa.com

**Region 2**—Jean Burns  
Universal Instruments  
607-779-7868  
burns@uic.com

**Region 3**—OPEN

**Region 4**—Chris FitzGibbon  
Orion Canada, Inc.  
613-563-9000  
chris@cyberus.com

**Region 5**—Joel Glazer  
Northrop Grumman ES  
410-765-2346  
joel\_glazer@md.northgrum.com

**Region 6**—Tom Gilchrist  
The Boeing Company  
425-234-4865  
tgilchrist9@attbi.com

**Region 7**—OPEN

**Region 8**—Michael S. Kiefel  
Abbott Laboratories  
614-624-7973  
Michael.kiefel@abbott.com

**Region 9**—OPEN

**Region 10**—Nancy Poma  
EDS  
248-265-0456  
nmpoma@comcast.net

**Region 11**—OPEN

**Region 12**—Irv Segal  
SysGen, Inc.  
847-205-5349  
Irv.segal@sysgeninc.com

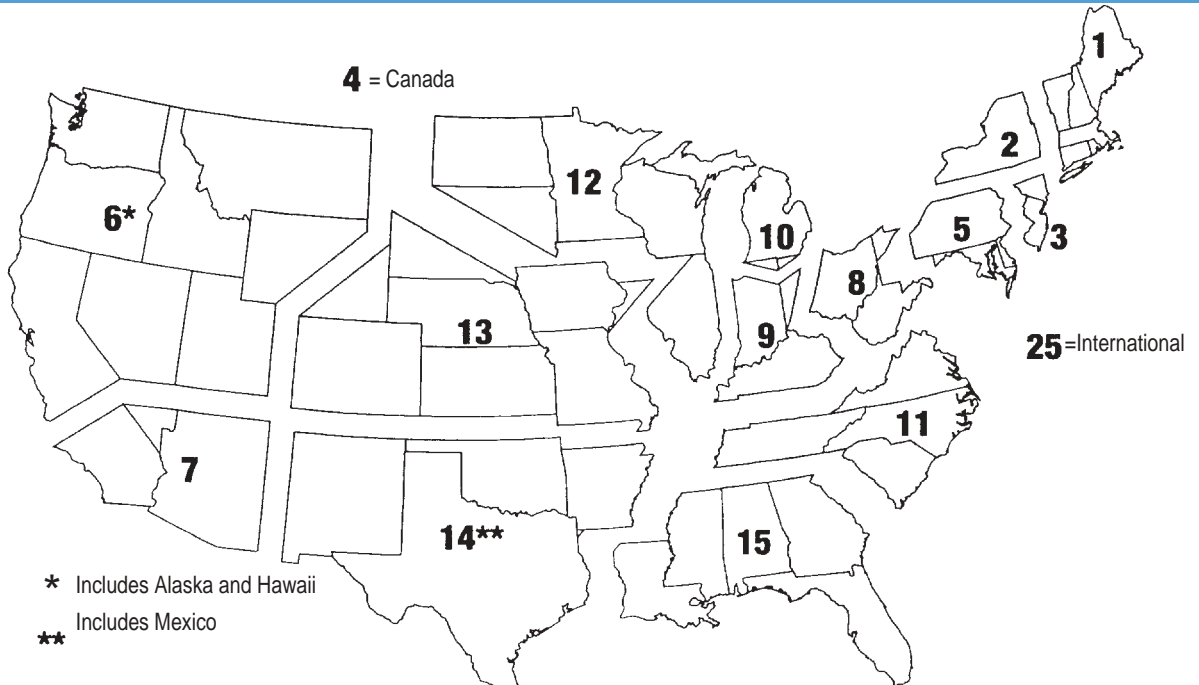
**Region 13**—Granville Jones  
JVJ Enterprises  
303-969-0228  
granville.jones@att.net

**Region 14**—W.L. 'Bill' Trest  
Lockheed Martin Aeronautics  
Company  
817-777-7598  
bill.l.trest@lmco.com

**Region 15**—Mark Neal  
Alcon Laboratories, Inc.  
407-384-1659  
mark.neal@alconlabs.com

**Region 25-International**—  
Zigmund Bluvband  
A.L.D. Ltd  
+972-3088-5100  
+972-5496-7201 cellular  
4+972-3977-5095  
zigmund@ald.co.il  
www.aldservice.com

## Regional Map



# SOFTWARE QUALITY

**SUBMIT ARTICLES FOR  
THE NEXT ISSUE OF  
SOFTWARE QUALITY BY  
MARCH 1, 2004.**

**DR. TOM F. GRIFFIN III**  
**PHONE: 334-244-3304**  
**FAX: 334-244-3792**  
**E-MAIL: TGRIFFIN@MAIL.AUM.EDU**

## EDITOR

DR. TOM F. GRIFFIN III  
AUM, IS & DS  
P.O. Box 244023  
Montgomery, AL 36124-4023  
voice: 334-244-3304 (Business)  
fax: 334-244-3792  
e-mail: tgriffin@mail.aum.edu

## EDITORIAL REVIEW BOARD

MICHAEL KRESS, Chair  
G. TIMOTHY SURRETT, Chair-Elect  
TOM GRIFFIN, Publications Chair

DAVE ZUBROW, Associate Editor  
LINDA WESTFALL, Associate Editor

## EDITORIAL POLICY

Unless otherwise stated, bylined articles, editorial commentary, and product and service descriptions reflect the author's or firm's opinion. Inclusion in *Software Quality* does not constitute endorsement by ASQ or the Software Division.

## ADVERTISING

FULL PAGE—\$500 per issue  
1/2 PAGE—\$250  
1/4 PAGE—\$125

# THIRD WORLD CONGRESS ON SOFTWARE QUALITY (3WCSQ)

**SEPTEMBER 26-30, 2005**

The planning for the Third World Congress for Software Quality (3WCSQ) is ongoing. It will be held in Munich, Germany, during the Oktoberfest (!), September 26-30, 2005. This congress is held every five years, and rotates among the American, European, and Japanese software quality organizations. The congress will last for five days and will consist of tutorials, workshops, panel discussions, paper and keynote sessions, excursions to the research laboratories of the main sponsors, and a five-day exhibition of software tool suppliers. There will be social events, receptions, and other opportunities to meet the experts and VIPs of the software engineering business. The Web site is [www.3wcsq.org](http://www.3wcsq.org). If you would like to submit a contribution to the conference, you can do so now via the Web site.

## KEY DATES

**Deadline for proposals of panels,  
workshops and tutorials**

October 15, 2004

**Deadline for full papers**

November 15, 2004

**Notification of acceptance**

February 15, 2005

**Final papers due** April 15, 2005

**Congress** September 26-30, 2005

The ASQ Software Division was the organizer and host of the inaugural congress that was held in San Francisco in June 1995. The Second World Congress (2WCSQ) was hosted by the Union of Japanese Scientists and Engineers (JUSE) and held in Japan in September 2000. The European Organization for Quality (EOQ)—Software Group is hosting the 2005 congress. The local sponsor is the ASQF.

Patricia McQuaid is the program chair for North, South, and Central America for the 3WCSQ, and did the same for the 2WCSQ in 2000. She can be reached at [pmcquaid@calpoly.edu](mailto:pmcquaid@calpoly.edu).

**Yes!** Please enter my subscription to *Software Quality Professional*, a quarterly publication focusing on the needs of professionals in software quality.

Member Number \_\_\_\_\_

### Subscriber information—please send to:

Name \_\_\_\_\_

Company Name/Agency \_\_\_\_\_

Title \_\_\_\_\_

Address \_\_\_\_\_ Apt./Suite # \_\_\_\_\_

City \_\_\_\_\_ State/Province \_\_\_\_\_

Zip+4/Postal Code \_\_\_\_\_ Country \_\_\_\_\_

Telephone ( ) \_\_\_\_\_ Fax ( ) \_\_\_\_\_

E-mail \_\_\_\_\_

### Payment options:

All orders must be paid in U.S. currency. Please make checks payable to ASQ. Checks and money orders must be drawn on a U.S. financial institution. All prices are subject to change without notice.

Payment enclosed:  Check  Money Order Amt. Paid \_\_\_\_\_

Please charge:  Visa  MasterCard  American Express

Charge Card No. \_\_\_\_\_ Exp. Date \_\_\_\_\_

Cardholder Name (please print) \_\_\_\_\_

Signature \_\_\_\_\_

Cardholder Address \_\_\_\_\_

City \_\_\_\_\_ State/Province \_\_\_\_\_

Zip+4/Postal Code \_\_\_\_\_ Country \_\_\_\_\_

Telephone ( ) \_\_\_\_\_ Fax ( ) \_\_\_\_\_

### Subscribe by:

Phone 800-248-1946 or 414-272-8575 (outside North America)

Fax 414-272-1734

Mail ASQ Customer Service Center, P.O. Box 3005, Milwaukee, WI 53201-3005

Online: <http://sqp.asq.org>

SOFTWARE QUALITY PROFESSIONAL			
	ASQ Members	Nonmembers	Institutional
U.S.	\$40.00	\$70.00	\$120.00
International	\$60.00	\$95.00	\$150.00
Canada	\$60.00	\$95.00	\$150.00

*Software Quality Professional* is published in December, March, June, and September.  
Subscription is for one year.

Priority Code: QRSADD1