

Software

QUALITY

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/

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

ARE WE DOING WELL OR ARE WE DOING POORLY?

BY LINDA WESTFALL

Abstract

Software metrics don't solve problems – people solve problems. What software metrics can do is provide information so you can make informed decisions and better choices. According to the new ISO/IEC 15939 Software Engineering—Software Measurement Process standard, decision criteria are the “thresholds, targets, or patterns used to determine the need for action or further investigation, or to describe the level of confidence in a given result.” In other words, you need decision criteria to obtain guidance that will help you interpret the measurement results. This article shows you how to establish useful decision criteria for different types of metrics.

Introduction

Measurement is common in our daily lives. We use measurement all of the time to help us make better decisions based on facts and information. For example, one of the first things you probably did today was measure. The alarm clock went off and you looked at the time. You Measured! Then you made a decision based on that measurement, “Do I have to get up right now or can I hit the snooze alarm and sleep for 10 more minutes?” That's the objective of software measurement as well, to provide software managers and software engineers with the information they need to make better decisions based on facts rather than on opinion or “gut feel.”

One of my favorite metric cartoons shows a metrics analyst presenting a variety of charts and graphs at a meeting. The caption has the boss asking, “Norman, please – just tell us, are we doing good, or are we doing bad?” In other words, unlike the simple decision of getting out of bed, in the complex arena of software engineering, it's not always so easy to interpret the software measurements and use them in decision-making. This is where defining the decision criteria for our metrics helps. We establish decision criteria to help us interpret the results of the measurement, then we use them in making our decisions.

According to Watts Humphrey, there are four major roles for software measurement [Humphrey 89]:

- **Control:** Control type metrics are used to monitor our software processes, products, and services and identify areas where corrective or management action is required.
- **Evaluate:** Evaluate type metrics are used in the decision-making process to study products, processes, or services in order to establish baselines and to determine if established standards, goals, and acceptance criteria are being met.
- **Understand:** As part of research studies, understand type metrics can be gathered to learn about our software processes, products, and services.

(cont. on p. 4)

CHAIR'S CORNER

BY MICHAEL P. KRESS

How Much Software Quality Do You Want?

An article in August's issue of *Quality Progress* entitled, "Widespread Effects of Defects" by Trudy Howles, assistant professor of computer science at Rochester Institute of Technology, insightfully reported the sorry state of quality of modern software products.

We are all painfully aware of the problem. Operating systems that do not perform as intended and lock up unpredictably, cell phones with aggravating flaws, business accounting programs with myriad bugs, and lackluster (or expensive or both) support resources. And yet we buy these products. Why, she asks, "Is it true that consumers would rather have it now and defective, or later and correct?"

Her analysis goes on to point to the very same lack of discipline management and process control that the aerospace industry, NASA, and the DOD discovered years ago. Lack of training, management oversight and involvement, lack of repeatable processes, project tracking, measurement, defect detection, prevention and appraisal of customer satisfaction, were the trademarks of the industry. Not surprisingly, these are the very reasons the DOD funded the creation of the Software Engineering Institute at Carnegie Mellon back in the 1980s. They are the very same reasons that the ISO community created ISO 9000:2000 with the strong emphasis on process definition, measurement, true management oversight, and customer satisfaction.

Howles cites the Standish Group study that reports cost over-



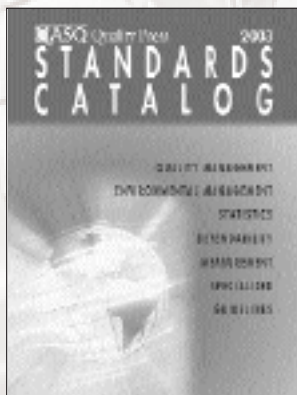
runs totaling 189% over estimates in 1994 have improved to 49% today. Wow! In 10 years, we have "improved" to "only" 50% over budget! That same group reported that the software industry spends more than \$250 billion a year in the United States on software, among which 53% goes into projects that are challenged (over budget, late, with less than full functionality) and 31% on projects that are impaired (cancelled). Only 16% is spent on projects that are successful – on time, within budget, meeting full functionality.

Exacerbating matters are initiatives like UCITA (Uniform Computer Information Transactions Act) which many critics feel improperly creates an imbalance in the protection of the interests of the consumer (licensee) in favor of the developer (licensor). While extremely contentious, UCITA is largely unpopular. The facts show that "...24 attorneys general, in letters to the National Conference of Commissioners on Uniform State Laws (NCCUSL), cite that UCITA would preempt important state-based consumer protections."¹ ASQ's Software Division has taken a similar position against UCITA in a letter to NCCUSL in July 1999. It is not my intent to open up debate or challenge the proponents of UCITA here, but rather to simply point to some of the reasons that software quality languishes today.

How you weigh in on this piece of legislation will depend on which side of the fence you are sitting. Are you a consumer or a developer? How good is good enough for deployment? What level of flaws are we willing to tolerate? Since some of us are both developers and consumers, but most of us are consumers, there has to be a compelling concern here for poor software quality. We can lament the sorry state of software quality, but only when the market forces of supply and demand for high quality software force developers to take action, will things improve.

¹ Accessed online March 1, 2000, at <http://www.arl.org/info/fm/copy/agoppltr.html>

New Quality Press Standards Catalog



Quality Press announces the publication of the latest Standards Catalog

featuring the complete portfolio of standards available for purchase from Quality Press.

Free copies are available to anyone who requests a copy. Order your copy through the Quality Press Bookstore (<http://qualitypress.asq.org>) or call ASQ at 800-248-1946 and request your copy today.



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. 17 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. Mandatory requirements employed and enforced to prescribe a disciplined uniform approach to software development are:
 - A. standards
 - B. procedures
 - C. processes
 - D. guidelines
2. When calculating cost of quality data, the cost of maintaining a Web site where customers can obtain service pack download would be considered:
 - A. a prevention cost.
 - B. an appraisal cost.
 - C. an internal failure cost.
 - D. an external failure cost.
3. Which of the following is an example of a test automation tool:
 - A. Requirements traceability analyzer
 - B. Capture and playback tool
 - C. Problem reporting tool
 - D. Code complexity analyzer
4. The major project management elements that must be balanced to have a successful project include all of the following EXCEPT:
 - A. cost.
 - B. schedule.
 - C. product.
 - D. process.
5. The Goal/Question/Metric paradigm is used to:
 - A. select metrics that align with the goals of the metric customer.
 - B. establish goals for implementing a metric program.
 - C. ensure that we have measurable goals established for our projects.
 - D. evaluate the usefulness of existing goals and metrics.
6. Which of the following are characteristics of a formal inspection?
 - I. Attended by project management
 - II. Attendees have assigned roles and responsibilities
 - III. Focuses on identifying work product defects
 - IV. Provides engineering analysis input into the creation of the work product
 - A. I and IV only
 - B. II and III only
 - C. II and IV only
 - D. II, III, and IV only

7. The baseline that is established when the system requirements are approved is the:
 - A. product baseline.
 - B. development baseline.
 - C. functional baseline.
 - D. allocated baseline

FDA COMPLIANCE CORNER

BY DAVID WALKER

This is a new column that will provide a concise update each quarter on computer validation subject matter. The goal of this column is to provide value to our Software Division members employed in FDA-regulated industries such as medical devices, laboratories, clinical research and development, and manufacturing of food, drugs, and cosmetics. I hope that others will start similar columns in the newsletter for other industry sectors.

FDA's Final Guidance for Scope and Application of Electronic Records and Signatures issued: www.fda.gov/cder/gmp/index.htm

August marks the sixth anniversary of the FDA's 21 CFR, Part 11; Electronic Records; Electronic Signatures. Through this time, there has been significant discussion between industry and FDA regarding its interpretation and implementation. The FDA is re-examining Part 11 in light of a new CGMP initiative "Pharmaceutical CGMP's for the 21st Century: A Risk-Based Approach; A Science and Risk-Based Approach to Product Quality Regulation Incorporating an Integrated Quality Systems Approach," www.fda.gov/oc/guidance/gmp/.html.

Note that all of the following have been withdrawn: 21 CFR, Part 11; Electronic Records; Electronic Signatures, Validation 21 CFR Part 11; Electronic Records; Electronic Signatures. Glossary of Terms 21 CFR Part 11; Electronic Records; Electronic Signatures, Time Stamps 21 CFR, Part 11; Electronic Records; Electronic Signatures, Maintenance of Electronic 21 CFR Part 11; Electronic Records; Electronic Signatures, Electronic Copies of Electronic Records CPG 7153.17: Enforcement Policy: 21 CFR Part 11; Electronic Records; and Electronic Signatures.

Although the Part 11 regulation remains in effect, it will be more narrowly interpreted by the FDA. The agency will exercise enforcement discretion with regard to certain Part 11 requirements. The message that stands out in the new Scope and Application guidance is "justified and documented risk assessment." Clearly we will be seeing more focus on "risk based approaches" to computer validation and, to our good fortune, this is not a new science. There is a wealth of literature including documented best practices, international standards, and even FDA guidance in the area of risk assessment and management.

The FDA anticipates initiating rulemaking to change Part 11 as a result of its reexamination in light of the new CGMP initiative. We will closely monitor any events of this nature in this column.

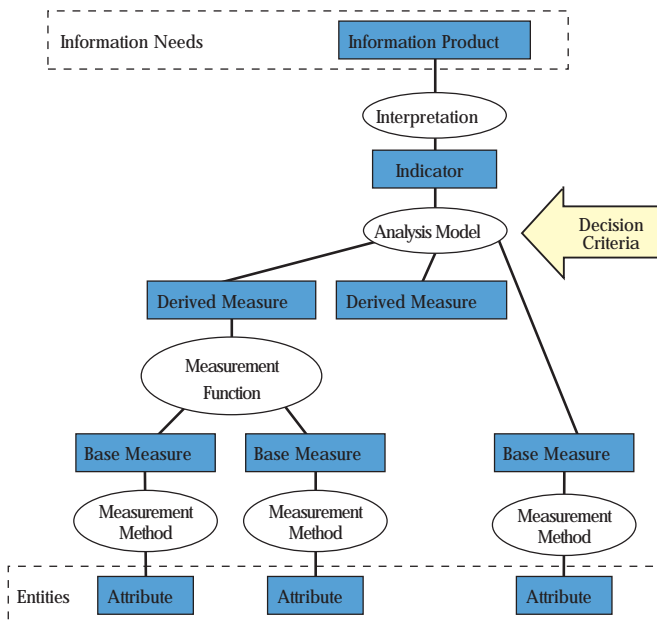


Figure 1. Measurement Information Model

- **Predict:** Predict type metrics are used to estimate the values of base or derived measures in the future.

We will use these different types of roles to explore establishing decision criteria for different types of measures.

Measurement Information Model

So where do decision criteria fit into the measurement process? To answer that, let's step back and look at the Measurement Information Model as defined in ISO/IEC and illustrated Figure 1. This model links the relevant entities and their attributes to an information product that matches the information needs of the measurement's user (customer).

Entities & Attributes: An entity is a specific object (e.g., software work product, software process or resource) that has specific measurable attributes (properties or characteristics). For example, if the software source code was the entity, we might want to measure the attributes of its size or the number of defects found in that code.

Measurement Methods & Base Measures: The measurement method includes the defined rules for mapping numbers or symbols onto the measured value to create a base measure for an attribute. For example, we might count semicolons to assign a base measure in logical lines of code to the size attribute for module X.

Measurement Functions & Derived Measures: A measurement function is the algorithm or calculation used to combine two or more base measures into a derived measure. For example, the measurement function of dividing the number of defects in module X by the number of lines of code in module X and multiplying by 1000 (to convert to defects per thousand lines of code) could be used to calculate the defect density for module X.

Analysis Model: According to ISO/IEC 15939, an analysis model is “an algorithm or calculation combining one or more base and/or derived measures with associated decision criteria. It is based on an understanding of, or assumptions about, the expected relationship between the component measures and/or their behaviors over time.” For example, based on historic data we could calculate the mean and standard deviation defect density for a set of source code modules in a similar product that met its post-release quality and reliability requirements and use those to model an acceptable range of defect densities for the current set of modules under consideration.

Decision Criteria: According to ISO/IEC 15939, “decision criteria may be calculated or based on a conceptual understanding of expected behavior. Decision criteria may be derived from historical data, plans, and heuristics, or computed as statistical control limits or statistical confidence limits.” For example, based on our historic model of “acceptable” defect density, we might establish decision criteria that state that any module with a defect density greater than one standard deviation above the mean is a candidate for reinspection before release to integration test. Any module with a defect density greater than two standard deviations above the mean would require further investigation as a candidate for possible reengineering.

Indicators, Interpretation, and Information Products: The application of the analysis model and decision criteria results in an indicator that “provides an estimate or evaluation of specified attributes derived from an analysis model with respect to the defined information needs.” One or more indicators, along with their interpretations, constitute an information product that addresses the information need. Figure 2 illustrates the indicator that might be used for this defect density example.

Decision Criteria for Control Type Metrics

Control type metrics are used to monitor our software processes, products, and services and identify areas where corrective or management action is required. Control type metrics act as “red-flags” to warn us when things are not as expected or planned. If all is going well, the decision should be “everything is fine and therefore no action is required.” The decision criteria for control type metrics usually take the form of:

- Thresholds
- Variances
- Control limits

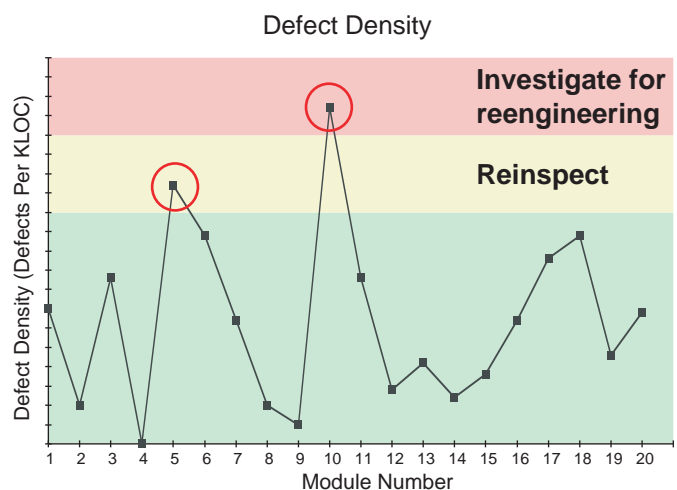


Figure 2. Example—Defect Density Indicator

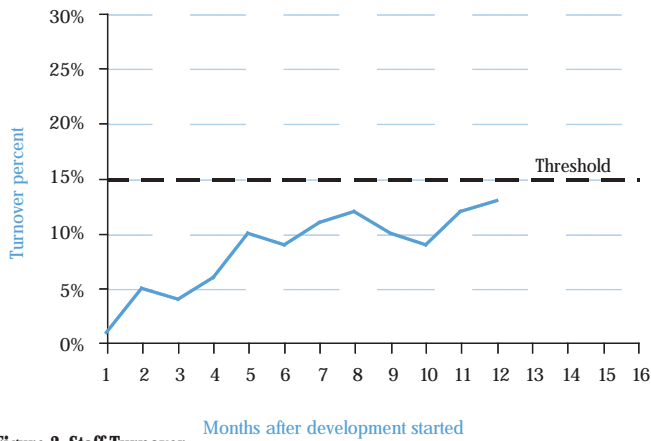


Figure 3. Staff Turnover

Thresholds: Thresholds are established boundaries that when crossed indicate that action or investigation is needed.

Thresholds may be established based on historic data like in the defect density metric discussed in the above example. In this example, two thresholds were established based on calculating the mean and standard deviation from historic defect density values. Similar graphs could be established for thresholds based on industry standards or benchmarks. For example, empirical evidence shows that modules with a McCabe's Cyclomatic Complexity greater than 10 are more error prone and harder to maintain [Jones 91]. We could create a graph similar to the one in Figure 2 where the thresholds are based on the acceptable values being a Cyclomatic Complexity ≤ 10 , the cautionary values being from say 10 to 20 and the unacceptable values being > 20 .

Threshold may be established based on future predictions. For example, if while planning our project we predict a 15% staff turnover rate, we will define our project's staffing and training plans accordingly. However, a risk exists that there may be a higher turnover rate than 15% that will impact our ability to complete the project on schedule and/or deliver the promised level of functionality or quality.

We can utilize this staff turnover measure as a risk management trigger. The decision criterion is therefore very straightforward, if the 15% threshold is exceeded; implement the staffing and training risk contingency plan. The run chart illustrated in Figure 3 could be used to track this measurement over time against its specified threshold.

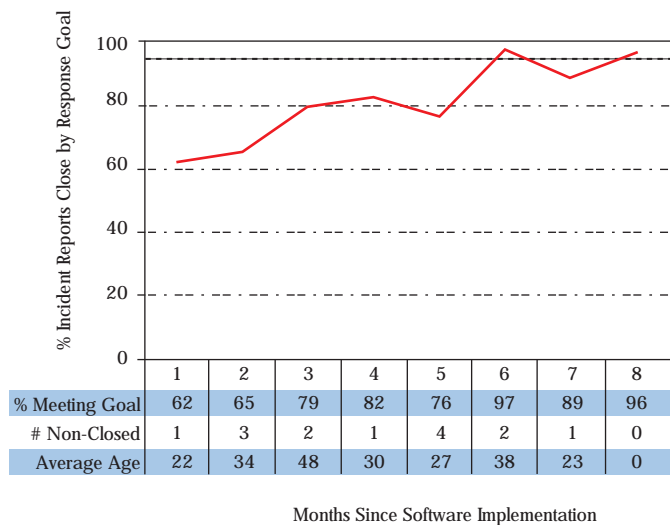


Figure 4. Problem Report Responsiveness

Similar graphs could be utilized to track other risk triggers based on predicted values used to plan the project. For example, we could track the actual requirements churn (scope creep) against predicted values, actual productivity level against the levels used to predict staffing requirements, or actual product size against the size estimates used to predict project effort and schedules.

Thresholds may be established based on customer requirements. For example, there may be customer requirements for capacity, throughput, response times, memory utilization, availability, reliability, or quality levels. These types of technical system requirements must be designed and built into the product. If the actual measures during modeling or testing do not meet the requirements (i.e., the threshold is passed) then a defect report should be opened to investigate the need for a product design change.

Another example of customer requirements being used as a threshold is illustrated in Figures 4 and 5. Here the customer had a service level agreement requirement that 95% of all major defects reported from the field would be corrected with 30 days. The run chart in Figure 4 tracks the percentage of problem reports closed each month within the 30-day limit. The problem with this graph was that it was a lagging indicator. The decision criteria for this measure was to apply more resources to problem resolution during the next month, which might help the subsequent month's measurement value if they were applied correctly; however, this does not correct the fact that we missed the requirement during the current period. We augmented this metric with the second distribution graph in Figure 5. This graph and its associated 30-day threshold allowed us to be more proactive in meeting our service level agreement requirements. The decision criteria for this age measure was to prioritize our defect resolution activities on defects that were approaching 30 days old.

Variances: Variances compare actual values with expected values and make decisions based on the magnitude of the difference. Many times the expected value is based on estimates or predictions. Variances are typically calculated in one of two ways, either as a ratio (actual value divided by expected value) or as an absolute delta (actual value minus expected value).

The decision criteria ranges for ratio type variances are typically established at 1 (or 100%) plus and minus a value representing an acceptable level of variation. Note that a vari-

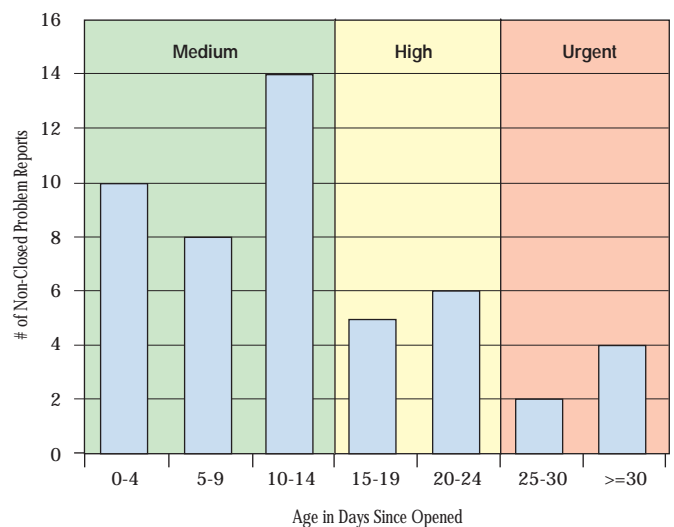


Figure 5. Problem Report Age

DOING WELL CONTINUED

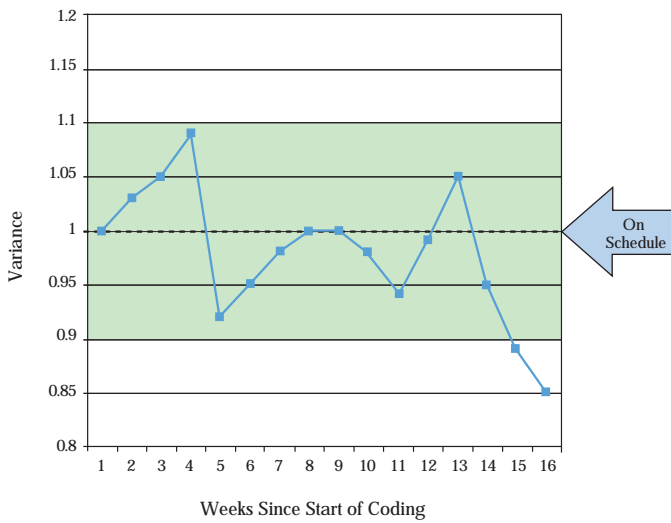


Figure 6. Assess Status of Coding Activities

ance of 1 (100%) indicates that the actual matches the expected value at that specific point in time.

The Assess Status of Coding Activities example in Annex A of the ISO/IEC 15939 standard uses a ratio type variance by dividing the Coding Units Planned to Date multiplied by 100 into the Progress to Date; where the Progress to Date is the sum of the percentage complete for each module. Figure 6 shows an example graph of what this ratio type variance measure might look like.

A ratio type variance is influenced by the changes in the magnitude of the actual and expected values over time. For example, if early in the project we compare the number of tasks actually completed to date vs. those estimated to be completed, each individual task carries a large impact on the variance. If we expect 10 tasks to be complete and one task is late, the variance is 9/10 or 90%. If we look at the variance later in the project, for example when we expect 100 tasks to be complete and one task is late, the variance is 99/100 or 99%. Depending on the magnitude of the estimate and actual at any given point in time, the same absolute delta between those two values will result in different ratio variances.

Early in a process or project it may also be more acceptable (i.e., less risky) to have larger variances because there is plenty of time left to make adjustments and bring the variance under control. But as the process or project nears its completion date the acceptable amount of variance may be much smaller.

For these two reasons, management may want to consider setting different variance ranges for different times in the process or project. For example, during a one-year-long project the initial acceptable variance might be set to $100\% \pm 25\%$ and the cautionary range set to $100\% \pm 40\%$. By mid-project, the acceptable variance might be reduced to $100\% \pm 15\%$ and the cautionary range reduced to $100\% \pm 25\%$. Finally, late in the project, acceptable variance might again be reduced to $100\% \pm 5\%$ and the cautionary range reduced to $100\% \pm 10\%$.

One of the advantages of using an absolute delta type variance is that it avoids the issue of the influence of the magnitude of the actual and expected values over time. Figure 7 shows an example of a resource utilization metric using absolute delta variances. Another advantage of using absolute delta variances is

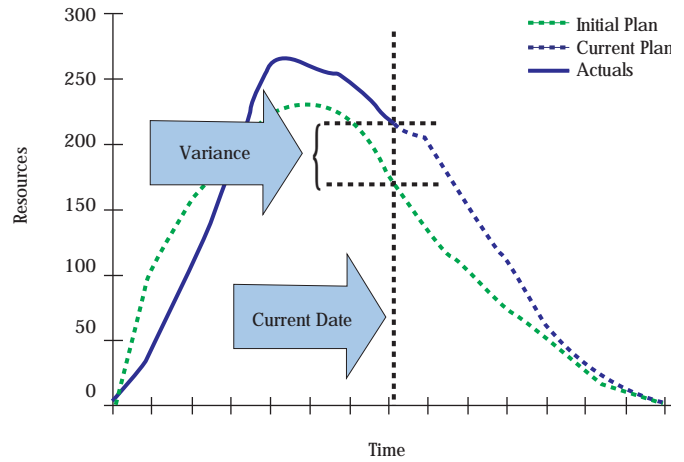


Figure 7. Resource Utilization

that for cumulative measures like tasks completed, effort expended, resources utilized or costs, the expected values typically planned out to the end of the process or project and the actuals to date can be extrapolated

Establishing decision criteria for a variance is typically a subjective judgment based on what management determines to be “in control.” Considerations when establishing these decision criteria include:

- **Historic Values:** Collecting and analyzing historic information on past variances for similar processes, products, or services can provide valuable information about the capabilities of the current estimation/prediction processes or what acceptable variances were historically when we had successful processes, products, or services. Current capabilities and past successes should be taken into consideration when establishing decision criteria.
- **Customer, Contractual, or Regulatory Requirements:** There may be specific customer, contractual, or regulatory requirements that dictate acceptable/cautionary/unacceptable values for certain variances.
- **Best Practices and Workmanship Standards:** Our organizations may also have established best practices or workmanship standards that dictate acceptable/cautionary/unacceptable values for certain variances.

If the measurement value for a variance is within the acceptable level, no action is required. If it is in the cautionary level it indicates that at a minimum the measure should be monitored more closely to ensure that it returns to an acceptable value. We may also want to do an analysis of the individual base or derived measures we are using to determine the cause(s) of their less than optimum values and take corrective action as necessary. If the measurement value for a variance is in the unacceptable range, an analysis must be conducted to determine the cause(s) and corrective action should be taken.

Control Limits: Statistical process control charts with control limits are one of the classic ways of controlling processes. To establish control limits, the mean and standard deviation are calculated from a set of sample data from the process. The mean becomes the center line (CL) and zones are created at ± 1 , ± 2 and ± 3 standard deviations from the mean. The upper control limit (UCL) is set at plus 3 standard deviations above the mean. The lower control limit (LCL) is set at minus 3 standard deviations below the mean (or at zero if this value is below zero and negative numbers are not valid measures).

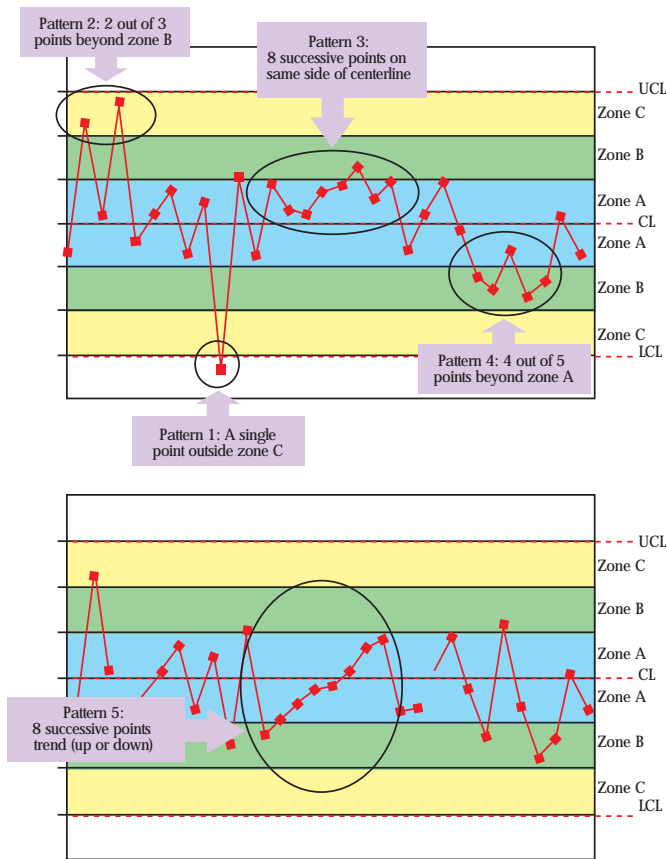


Figure 8. Control Charts Decision Criteria Patterns

As illustrated in Figure 8, the decision criteria is based on five patterns:

- Pattern 1: A single data point more than ± 3 standard deviations from the mean
- Pattern 2: Two out of three data points in a row more than ± 2 standard deviations from the mean
- Pattern 3: Eight successive data points on the same side of the centerline
- Pattern 4: Four out of five data points in a row more than ± 1 standard deviations from the mean
- Pattern 5: A run of eight successive data points in a row either all trending up or all trending down.

While these data patterns are all possible they are highly improbable. If one of these patterns is identified, the process that created the data should be investigated to determine if there is an assignable cause that needs to be corrected or if the data pattern is the result of normal variation in the process.

Decision Criteria for Evaluate Type Metrics

Evaluate type metrics are used to examine and analyze our software processes, products, and services as part of the decision-making process. Examples of using evaluate type metrics include:

- Performing cost/benefit analysis
- Analyzing and prioritizing choices
- Performing analysis to determine if a process is ready to start or end (entry and exit criteria)

Cost/Benefit Analysis: Cost/benefit analysis looks at the predicted costs of a project or action and compares those

against the predicted benefits of undertaking that project or action in order to:

- Determine if predicted return on investment (ROI) is sufficient enough to recommend initiating that project or action
- Select between alternative projects or actions

Typically the cost/benefit analysis is performed using a ratio calculated by dividing the benefit of a project or action by its cost. Examples of cost include the cost of people (salaries, benefits, travel, training, contractor's fees), materials, methods and tools, capital equipment, infrastructure, and risks. Examples of benefits include future revenue, maintaining or increasing market share, improving technical leadership, increasing capability or capacity, and reducing risk.

Any benefit-to-cost ratio less than 1 indicates that the project or activity costs more than its expected benefit and should be rejected. However, we also need to make a profit on our investment, a benefit to cost ratio of 1 indicates only a breakeven. When establishing the decision criteria for a benefit-to-cost ratio we need to consider an acceptable profit margin. For example, if we determine that we need to make at least a 15% profit on our investment or the money should be used elsewhere then we would establish our decision criteria for the benefit-to-cost ratio at ≥ 1.15 .

As a further example, consider a project risk that we have analyzed and determined to have a 25% likelihood of turning into a problem (the probability of an undesirable outcome) and if it turns into a problem we are predicting the loss will be \$300,000. The expected value of the risk, its Risk Exposure (RE), is therefore 25% of \$300,000 = \$75,000. If our decision criterion for Risk Exposure indicates that if our exposure is over \$50,000 for any given risk, we will explore risk reduction alternatives to reduce that risk. As illustrated in Table 1, we have come up with three alternatives to consider.

Alternative 1 is expected to reduce the risk's probability of turning into a problem to 5%. If we implement this alternative, our new predicted risk exposure is now 5% of \$300,000 = \$15,000. This is a risk reduction benefit of \$75,000 - \$15,000 = \$60,000. But this risk reduction activity is predicted to cost \$65 so the benefit to cost ratio is $\$60,000 / \$65,000 = .9$. Since this is less than 1, we reject this alternative.

Alternative 2 is expected to reduce the risk loss if it turns into a problem to \$160,000. If we implement this alternative, our new predicted risk exposure is now 25% of \$160,000 = \$40,000. This is a risk reduction benefit of \$75,000 - \$40,000 = \$35,000. This risk reduction activity is predicted to cost \$25,000 so the benefit to cost ratio is $\$35,000 / \$25,000 = 1.4$. Since this is less than our decision criteria of 1.15, this choice is still in the running.

Alternative 3 is expected to reduce the risk's probability of turning into a problem to 20%. If we implement this alternative, our new predicted risk exposure is now 20% of \$300,000 = \$60,000. This is a risk reduction benefit of \$75,000 - \$60,000 = \$15,000. This risk reduction activity is predicted to cost \$2,000 so the benefit to cost ratio is $\$15,000 / \$2,000 = 7.5$. Clearly this alternative has the highest benefit to cost ratio of the three alternatives. But wait – remember our risk exposure decision criteria that said any risk exposure over \$50,000 requires us to explore risk reduction alternatives to reduce that risk? The new risk exposure for this risk after the alternative 3 risk reduction action is taken is still at \$60,000. Therefore, unless we implement both alternatives 2 and 3, we must select alternative 2.

(cont. on p. 8)

Risk #	Prob(UO) before	Loss (UO) before	RE before
143	25%	\$300K	\$75K

Alternative	Prob(UO) after	Loss(UO)after	RE after	Cost	RRL
1	5%	\$300K	\$15K	\$65K	0.9
2	25%	\$160K	\$40K	\$25K	1.4
3	20%	\$300K	\$60K	\$2K	7.5

Table 1. Risk Reduction Leverage

Note that if we choose to implement alternative 2, our risk exposure is still \$40,000. Therefore, \$40,000 would be counted as a risk cost if we were calculating the cost of this project as part of a cost/benefit analysis to determine if we should initiate it.

Analyzing & Prioritizing Choices: When analyzing and prioritizing choices, measurements are utilized to rank order a set of items. We have already discussed two examples of this. First, in our problem report age discussion we established decision criteria that prioritized problem resolution priorities based on the age of the problem report. Second, in our discussion on cost/benefit analysis we established decision criteria that prioritized the selection between risk reduction alternatives and the resulting risk reduction leverage (RRL).

One classic example of using measurements to analyze and prioritize is to create a Pareto chart of a set of data. Pareto analysis is the process of ranking problems or categories based on their frequency of occurrence or the size of their impact in order to determine which of many possible opportunities to pursue first. Pareto analysis is based on the Pareto principle (80% of the trouble comes from 20% of the problems).

A Pareto chart is a bar chart where the height of each bar indicates the frequency or impact of problems. To construct a Pareto chart:

1. Determine the categories for the chart. For example, we might decide to examine the number of problems reported by module. Other examples might include problems by root cause, by phase introduced, or by reporting customer.
2. Select a time interval for analysis. For example, we could look at all defects reported in the last six months.
3. Determine the total occurrences for each category. For example, the number of defects per thousand lines of code (KLOC).
4. Rank order the categories from the largest total occurrences to the smallest and create a bar chart where the bars are arranged in descending order of height from left to right.

For example, as illustrated in Figure 9 we could create a Pareto chart of modules ranked by their defect density to prioritize those modules for reengineering or for additional defect detection effort.

Another method for analyzing and prioritizing items is to create a scorecard. Figure 10 shows an example of a scorecard for ranking and choosing suppliers. The decision criteria for this scorecard is to simply choose the supplier with the highest score.

Entry & Exit Criteria: Entry/exit criteria are specific, mea-

asurable conditions that must be met before the process can be started/completed. The decision criteria for one or more evaluation type metrics are frequently used as entry and/or exit criteria for a process.

For example, if we are evaluating whether or not system testing is complete, we might look at measures including:

- System Test Effort Variance decision criteria (see Figure 11): Cumulative system test effort rate matches plan within a 10% variance
- System Test Activity Status decision criteria (see Figure 12):
 - 95% of all planned test cases executed
 - No blocked test cases
 - 95% of all test cases executed were passed
- Problem report arrival rate decision criteria (see Figure 13):
 - Critical problem report arrival rate curve slope approaching zero with no new critical problems reported in last week of testing
 - Major problem report arrival rate curve slope approaching zero with no new major problems reported in last two days of testing
 - Minor problem report cumulative arrival rate curve slope approaching zero
- Non-closed problem report counts decision criteria (see Figure 13):
 - Zero non-closed critical problem reports
 - Less than 10 non-closed major problem reports all with customer approved work-arounds
 - Less than 25 non-closed minor problem reports

Decision Criteria for Understand & Predict Type Metrics

Understand type metrics are used as part of research processes to learn about our software processes, products, and services. Examples of using understand type metrics would be to determine:

- How much are we spending on software development? On testing?
- Where do we allocate and use resources throughout the life cycle?
- How many errors and of what types are typical in our software products? How much do they cost us?
- What are our current productivity levels?

The information from understand type metrics can be used to:

- Establish baselines, standards, and goals
- Derive models of our software processes
- Examine relationships among process parameters

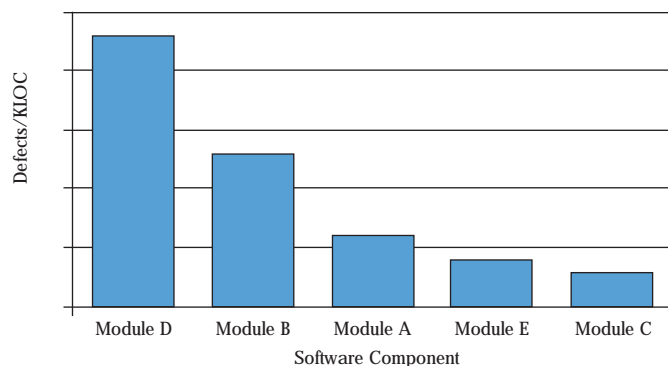


Figure 9. Pareto Chart—Defect Prone Modules

- Target process and product improvement efforts
- Better estimate project effort, costs, and schedules

This information can then be utilized to help establish the decision used for other types of metrics. For example in our defect density discussion earlier, we had to understand the average and standard deviation for a historic set of defect density data in order to establish the thresholds used as decision criteria for controlling the defect density of the modules in our current project.

Predict type metrics are used to estimate future values of base or derived measures. Predict type metrics are used to answer questions like:

- How much will it cost? (Budget)
- How long will it take? (Schedule)
- How much effort will it take? (Staffing)
- What other resources will it take? (Resources)
- What is the probability that something will go wrong? And what will it cost if it does? (Risk)
- How many errors will it have? (Quality)
- How will those errors impact operations? (Reliability)

Going back to the ISO/IEC 15939 Software Engineering—Software Measurement Process standard definition, decision criteria are the “thresholds, targets, or patterns used to determine the need for action or further investigation, or to describe the level of confidence in a given result.” It is typically the “level of confidence in a given result” portion of this definition that applies to both understand and predict type metrics.

Considerations when determining our confidence level in a given result include:

- The completeness of the data used. For example, if we are trying to understand the amount of effort we expend on software development does our time reporting system include all the time spent including unpaid overtime?
- Are the data used subjective or objective? Has human or other types of bias crept into the data?
- What is the integrity and accuracy of the data? For example, if we again consider timecard data are people completing their timecards as they complete tasks or are they waiting until the end of the week and then estimating how much time they spent on various projects.

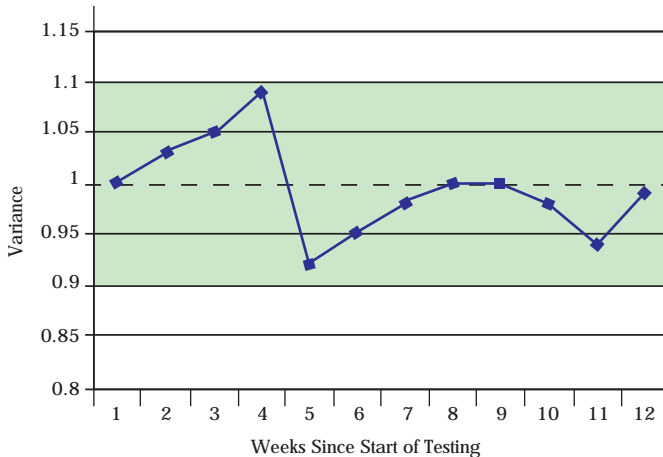


Figure 11. System Test Effort Variance

Scorecard for Ranking Potential Suppliers

Attribute	Max Score	Supplier 1	Supplier 2	Supplier 3
Ability to deliver by date needed	10	10	7	8
Purchase price / licensing costs	10	7	5	10
Licensing restrictions	5	5	4	5
Operating costs	15	12	15	5
Maintenance costs	10	5	10	7
Process capability	10	10	8	5
Product functionality matches needs	20	18	16	8
Product quality	20	20	15	15
Ease of integration with existing systems	5	3	5	3
Ease of integration with our business processes	10	10	7	10
Ability to customize product	5	5	4	5
Technical support	5	5	3	2
Training availability	10	10	5	5
Total Score	135	120	104	88

Ability to deliver by date needed = 10 points minus one point for each week past needed date

Product functionality meets needs = (# requirements met/ Total requirements) x 20

Figure 10. Supplier Selection Scorecard

- How stable is the product process or service being measured? For example, if we are using a new process or a new programming language, we may not be very confident in a prediction we make based on our historic data, or we may not have any relevant historic data upon which to base our predictions.
- What is the variation within the data set? For example, Figure 14 shows the distribution of the responses to three different questions on a customer satisfaction survey. For each of these questions the average customer satisfaction level is 3. But how confident are we that we “understand” that the confidence level is 3 for each of these questions? If we have a distribution like the one for Question A, we can be confident that 3 represents our customer’s opinions. However we would have very little confidence that a satisfaction level of 3 truly represented our customer’s opinions if the response distributions looked like the one for questions B or C.

Conclusions

I believe that having clearly defined and appropriate decision criteria are important in ensuring that our measurement indicators are correctly interpreted and applied by our measurement users. While the new ISO/IEC 15504 Software Measurement

(cont. on p. 10)

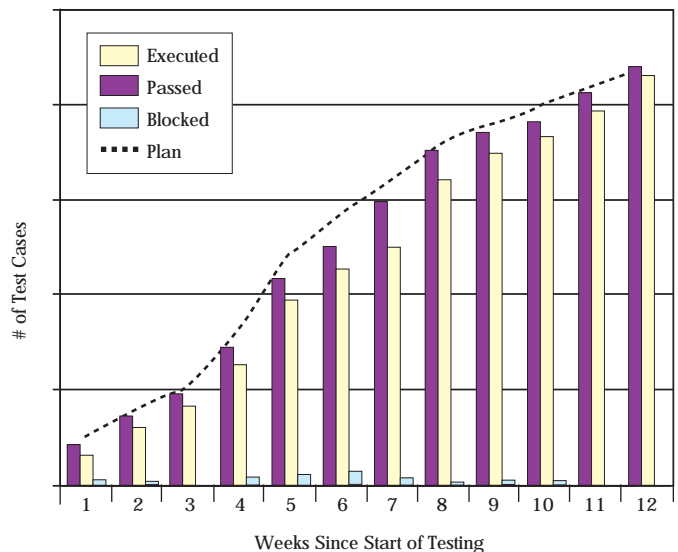


Figure 12. System Test Activity Status

Process standard defines what decision criteria are and gives a few examples, it doesn't provide much "how to" guidance to help us establish decision criteria for our metrics. It is important to decide how we are going to use the metrics we are planning to collect and whether we need them to control, evaluate, understand, predict, or a combination of those roles. This creates more discipline and direction in helping us determine meaningful decision criteria. There is a magnitude of ways that collected data can be compiled and presented, but how they are interpreted must be guided through careful establishment and definition of the decision criteria for each type of metric. It is my hope that the examples and discussion in this article will provide a starting point for further discussion on how to approach what role the data are to play and how to define and establish decision criteria for different types of metrics.

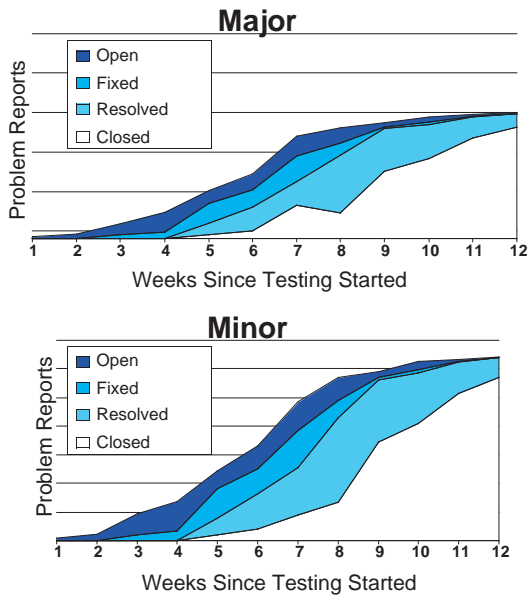
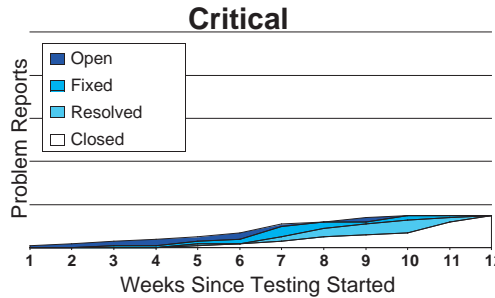


Figure 13. Cumulative Problem Report Arrival Rates by Status

Cumulative Problem Report Arrival Rates by Status



Question Response Distribution Report for Current Satisfaction Levels

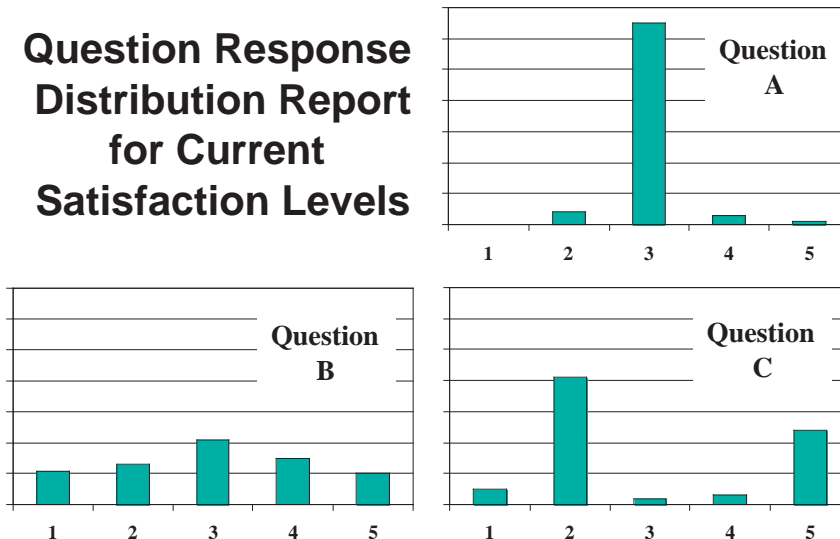


Figure 14. Customer Satisfaction Survey Results

References

Humphrey, Watts S.; *Managing the Software Process*; Addison-Wesley Publishing Company, Reading, MA; 1989. ISBN 0-201-18095-2.

International Standard, *Software Engineering—Software Measurement Process*, First edition 2002-07-15.

Jones, Capers, *Applied Software Measurement: Assuring Productivity and Quality*, McGraw Hill, New York, 1991. ISBN 0-07-032813-7.

McGarry, John; David Card; Cheryl Jones; Beth Layman; Elizabeth Clark; Joseph Dean; and Fred Hall; *Practical Software Measurement: Objective Information for Decision Makers*, Addison-Wesley, Boston, 2001, ISBN-0-201-71516-3.

Additional Resources

Visit The Westfall Team's Software Metrics, Measures, and Analytical Methods Library & Information page at http://www.westfallteam.com/software_metrics_measurement_analytical_methods.htm for more papers and information including a download for a Metrics Report Definition Template.

Linda Westfall is president of The Westfall Team, which provides software metrics and software quality engineering consulting and training services. Prior to starting her own business, she was the senior manager of quality metrics and analysis at DSC Communications where her team designed and implemented a corporate-wide metric program. She has more than 20 years of experience in real-time software engineering, quality, and metrics.

Very active professionally, Westfall is past chair of the ASQ Software Division. She is an ASQ certified software quality engineer and certified quality auditor and is a professional engineer in software engineering in the state of Texas.

She may be reached at lwestfall@westfallteam.com.

PERCEPTIONS OF THE IMPACT OF ISO 9001:2000 ON ORGANIZATIONS

BY EUGENE (GENE) M. BARKER

Much has been written regarding the challenges of developing and implementing a quality management system that complies with ISO 9001:2000. Some of the concern involves the practical and procedural aspects of changing from alignment to 20 “shall-based” elements to the development of a system composed of business process that assure quality throughout the entire product life cycle.

The impact of the changes on an organization is a function of the effectiveness of meeting the three primary aspects of the standard: management involvement, customer satisfaction, and process management. The ability to integrate these three factors successfully into a system will decide the gains from the new system. At the base of the year 2000 revision is an understanding that it is no longer a collection of requirements; rather it is a specification of an integrated system. Every factor has the ability to enhance or detract from the total performance of the system. And the allocation of resources to those areas having the highest leverage will assure an effective quality management system that is also efficient.

While ISO 9001 has always talked about a system for the management of quality, only the most advanced organizations implemented the requirements into an integrated system. The more typical approach was to identify each of the “shalls” within the standard, most frequently grouped within one of the 20 elements, and then set about defining the approach to address each “shall.” The result was a collection of quality practices that in aggregate met the standard, but did very little to improve the efficiency and effectiveness of the organization.

The year 2000 revision of ISO 9001 offers several opportunities for improved business performance, but only if compliance is achieved by the implementation of a total system. This starts with the involvement of senior management and their commitment that they will use the data from the customer satisfaction measures and the process metrics to drive continuous improvement. This is a management system with a quality focus and to be effective must be owned by the senior management team.

The process of defining the system should begin at the end with an understanding of your customers and what they need to be satisfied. This is not a simple step. It requires measurement and measurement requires communication with the customer. It is very easy to believe that you know better than the customers what they need. If the organization takes the time to understand its customers’ needs, systematically measures its performance in achieving these needs, and then implements actions to improve these measurements, customer satisfaction will improve. It is this focus on customer satisfaction that forms the foundation of the new standard.

Once the customer’s needs are understood then the easiest way to assure that the entire organization is focused upon meeting these needs at the lowest cost is through the identification of processes. Every process has an input, a series of activities that add value, and an output. This output becomes the input for another process or results in a product or service that is delivered to an external customer. The business system is this collection of processes.

Every process requires a minimum of five measures, perhaps more.

- The quality of the output must be measured in terms of the customer that receives the product or service from the process performer
- The internal quality of the process is measured to identify all waste caused by doing things over again and checking to assure that things were done correctly. These are non-value added activities and provide opportunities for improvement.
- The other attribute of interest to the customer is the on-time delivery of the product or service, frequently called compliance to schedule.
- And internally it is important to know the amount of time that is required to perform the process, the cycle time.
- The final essential measurement is cost per unit of output.

It is essential that these measures be carefully selected. The members of the organization will attempt to improve the performance of these measures, and if the wrong thing is measured the wrong thing will get better. Never underestimate the tried and true statement, “What gets measured gets worked.”

Improvement doesn’t just happen, at least not significant change. That is the advantage of the process approach to management. One is able to understand the interaction of the various processes and work to optimize the overall system. And the only way that improvement can be maintained is to change the process. Once the process is revised this new approach will be captured.

Every process should have at least one aggressive improvement goal. And by aggressive I mean a 50% reduction or the doubling of output per unit of cost. Small goals result in people working harder; aggressive goals result in people working smarter to revise the process.

Now that you understand your customer’s satisfaction and your processes you are ready to step back and look at the entire system. This is the first time that I have suggested that you look at the ISO 9001 standard. If you have done the above well you will be in compliance with the standard. Yes, you may need to document your processes or take care in retaining the few records that are needed to operate the business, but if you were diligent in defining the necessary processes and their inherent waste you are 90% of the way to having a compliant quality system.

Use the standard to identify opportunities that you may have missed. Challenge those requirements within the standard that make no business sense. Do not be a slave to the standard. My experience is that in almost every case when a requirement is considered to be unreasonable it is a result of not clearly understanding the benefit of the requirement. And once that is understood, the related process can be improved to achieve the objective.

If you are successful in doing the above, your business will be healthier and your customers happier. That means lower costs and increased revenue, a winning formula for assuring business success. The biggest benefactors will be your employees. People enjoy working in an organized and efficient organization where everything is done right the first time. And they will be proud of the products or services they provide and of the organization.

Eugene M. Barker is a technical fellow at The Boeing Company responsible for quality industry association interfaces. He led the industry writing team that drafted SAE AS9000 and chaired Working Group 11 of ISO TC20 and the International Aerospace Quality Group (IAQG) that developed AS/EN9100. He is a Fellow of the American Society for Quality, a member of the Registrar Accreditation Board (RAB) board of directors, and a founding member of the IAQG.

SYNOPSIS OF THE 4TH INTERNATIONAL CONFERENCE ON SOFTWARE TESTING (4TH ICST®)

APRIL 2-4, 2003
COLOGNE, GERMANY

BY MICHAEL P. KRESS

Abstract

The ICST® is held annually in conjunction with the Software Quality Management Conference (SCM®) and the Conference on Software Validation for Health Care (CSVHC®). It attracts more than 760 participants from 20 countries. It is the largest European forum for the discussion of current topics in software quality and testing. As a representative of American interests as part of the Americas Aerospace Quality Group, I attended in hopes of learning and understanding the European perspective on software quality so that the work of organizations such as ASQ and AAQG can one day be harmonized with international standards where practical. The proceedings are more than 500 pages long. This synopsis summarizes the highlights of the sessions that I attended.

Using SPICE (ISO/IEC 15504) self-assessments for improving the test process.

Ole Fischer
Ericsson Eurolab,
Herzogenrath/Germany

Ericsson conducted two SPICE assessments in 2002 on software testing and system integration and testing. They claim the results place both processes on the SPICE scale at level 3.

SPICE stands for Software Process Improvement and Capability dEtermination. Unlike the original CMM, the SPICE model provides more granularity to the assessment criteria and includes one more level (level 0 - Incomplete). The levels are:

- L5—Optimizing
- L4—Predictable
- L3—Established
- L2—Managed
- L1—Performed
- L0—Incomplete

The model groups 36 processes into five process categories:

- CUS: Customer-Supplier
- ENG: Engineering
- SUP: Support
- MAN: Management
- ORG: Organization

To obtain a rating, nine process attributes were rated as N - Not achieved, P - Partially achieved, L - Largely achieved, F - Fully achieved. The process attributes are:

- Continuous improvement
- Process innovation

- Process control
- Process measurement
- Process deployment
- Process definition
- Work product management
- Performance management
- Process performance

Ericsson compares the benefits of SPICE assessments to ISO audits as follows:

SPICE	ISO
Large document	small document
Detailed model	abstract model
Designed for sw	designed for mfg.
Sw focused	no software focus
Six capability levels	pass/fail only
Thorough	brief
Positive evidence	negative evidence
Fact finding	fault finding
Cooperative	adversarial
Open	defensive
Group oriented	individuals

The conclusions drawn by Ericsson are that the SPICE model provides more detailed results than standard audits and that it can be extended and adapted to suit business needs.

Author's note: What they did not present is the cost of conducting the SPICE assessment and the ROI. They concluded that ISO audits don't tell much about software capability. This is true. However, the newly released AS9006 supplement to AS9100A, while not being as detailed as SPICE, provides a reasonable and effective evaluation of an aerospace software environment that would otherwise be ignored or wrongfully certified if based only on ISO 9001:2000. Data from the SEI states that,

"Because there are practices in the CMM that are not addressed in ISO 9000, it is possible for a level 1 organization to receive 9001 registration."

—Mark C. Paulk, "A Comparison of ISO 9001 and the Capability Maturity Model for Software," Software Engineering Institute, CMU/SEI-94-TR July 12, 1994.

My point is this. There is a heavy cost associated with SPICE/CMMi assessments. It may be overkill for some organizations. On the other hand, there is the wrong impression that pure ISO 9001 audits adequately cover software. This is underkill. AS9006 was designed to provide an adequate evaluation of an aerospace software environment without the pricey consequences of SPICE.

Extreme Programming and Testing

Kent Beck
Three Rivers Institute

Kent Beck is appropriately regarded as the pioneer and champion of extreme programming. Simply put, extreme programming is a lightweight process for developing software that is based on decomposition of the project into many small projects, depends on rapid feedback, nearly concurrent testing and coding, and lean methods. The payback is claimed to be reduced waste and predictability. Some of the non-legacy techniques that are part of extreme programming are:

- Rapid feedback
- Incremental change
- Unit testing
- Pair programming
- Refactoring
- On-site customer support
- Frequent releases
- Collective code ownership

Beck claims the technique is not his invention, but was “stolen” from the manufacturing industry. Although he didn’t specifically mention it, concepts such as lean and just-in-time are common to the technique. Just as “just-in-time” reduces inventory, so does extreme programming. Excessive inventory leads to waste, errors, and obsolescence. Re-factoring is simply evaluating the code for duplicity and redundancy. “Pair programming” is simply concurrent code review. Collective code ownership allows anyone to refactor the code at any time.

The ultimate claimed benefit is a very short cycle time between production and revenue. One question came up as to how you get the customer involved in such short-term feedback loops. The reply is that it requires a culture change.

Author comment: I had the opportunity to chat at length with Beck. The benefits of XP are, as yet, largely unproven. No data were presented on case histories using the technique. While the objectives of XP are highly commendable, (reduced waste, early time to revenue) there are no hard data to show that such objectives can be gained by “shortcutting” established methods.

Test Documentation From the Auditor’s Perspective

Werner Schmidgruber DGR Wirtschaftsprüfung, Bonn

The author conducts audits within the banking and IT industry to German standards, IDW PS330 AND IDW PS 880. Audits are characterized by four basic steps:

- Test and audit requirements definition
- Test preparation
- Test execution and controlling
- Test documentation

The first step determines the rules for testing the requirements, reviewing the requirements and measuring the completeness and consistency. Test program metrics include:

- Number of function points,
- Number of executed test procedures,
- Number of total defects found vs. the number of procedures,
- Number of problem reports by priority.

Defect tracking and configuration management are also assessed concurrently. Test cases are evaluated for dependencies, priorities, targets, representation of the test environment, description of the input and expected test result. Results must include test case, name of tester, date and time, description of the inputs and outputs. Error handling and documentation are also closely examined. Curiously, not much was discussed on configuration control of the software product under test. This is a common source of error in testing in the United States. I questioned Schmidgruber about the possible relationship of these audits to the audits conducted by the TickIT program and against ISO 9001/9000-3. Apparently there is not much correlation; however, some credit is given for ISO recognition.

Extreme Programming Considered Harmful for Reliable Software Development

Gerold Keefer, AVOCA

Keefer portrayed extreme programming as just another “Holy Grail” or flavor-of-the-day, along with structured programming, 4GL, CASE, and Agile processes. He characterized it as heavy on exaggeration and hype, while conceding that it is better than “ad hoc.” He listed some of the flaws, as he perceived them:

- Testing can only find defects when they exist, and cannot confirm the absence of bugs.
- A one-room setting does not promote productivity.
- On-site customer is not practical; customer reps change, promoting inconsistency.
- Pair programming – No evidence of superiority.
- Collective code ownership is infeasible and unhelpful.

A spirited discussion ensued afterward. I commented that it is unwise to throw the baby out with the bathwater. While the technique might not be rigorously defended as yet with overwhelming data, the concepts of lean, early feedback, reduction of useless documentation, and generally a shorter life cycle are commendable goals. It remains to be seen if post-deployment reliability and user satisfaction are truly met by extreme programming.

Software Reliability Over the Life Cycle: Expectations and Reality in the Automotive Industry

Joachim Hauser, BMW (D)

Hauser’s presentation was intriguing not only because he chronicled the growth of software controlled systems in automobiles, but also because of the similarity of controls used in the aerospace industry. Automobiles supported by software began in the ‘70s with electronic ignition, speed control, door locks, and various check controls. Over the next two decades, gear control, air conditioning, ABS, navigation systems, CD changer, airbags, roll stabilization, Internet portal, 42-volt systems, and car office, among many other systems, have materialized. By the year 2010, systems are envisioned for steer-by-wire, brake-by-wire, automatic parking, communication between cars, connected drive, track control, and a host of other acronym-identified systems that I was not fast enough to capture with my notes. The amount of information in flash memory has increased from 10KB in the early ‘80s to 10 MB today and is forecast to be a gigabyte by the year 2010.

Are software faults important? You bet. One report described the recall of 110,000 sport utility vehicles because of a software glitch in the car’s cruise control system. It reported one British motorist died when his car went out of control and several U.S. motorists reported similar control problems. After much investigation the cause was traced to a programming error in a chip in the power train control module.

At BMW, software development and test follow the well-known classical V-model used by many aerospace organizations. Software design is an open and modular architecture that uses hardware-independent software layers. Modularity allows independence of components and limits interactions. The objective is controlled software components as well as hardware components.

Interestingly and commendably, BMW requires all software suppliers to follow the obligatory process standard SPICE

(cont. on p. 14)

(ISO/IEC 15504). A system of process attributes is measured for maturity. For example,

- Project management
- System requirements analysis
- Software requirements and analysis
- Software design
- Software construction
- Software integration
- Software test
- Configuration management
- Quality assurance
- Problem reporting
- Supplier management

are process attributes that are ranked by the degree to which these are implemented (fully, largely, partially, and not satisfied) and charted against a capability level (0-5).

Formal Verification Methods are used to test specifications using CASE tools. This reduces development time and makes complete verification and validation of the specification possible. BMW's current "7-Series" electronic systems are comprised of approximately 65 ECUs, 5 bus systems, and over 50 MB of software. In summary, Hauser concluded by stating that proactive software design and development processes are required to ensure the quality and stability of networked functions, which is required to guarantee success.

Strategic Planning of QA Activities for Safety Critical Software

Dr. Bettina Buth

Dr. Buth began her presentation by emphasizing the fact that QA activities within development projects are always constrained by cost and time restrictions. The focus of her message is that the skillful use of the combined techniques of failure modes and effects analysis (FMEA) and fault tree analysis (FTA) can lead to the most effective use of valuable test and evaluation effort.

Citing safety critical specifications such as D0-178B for aviation and CENELEC 50128, the European standard for railway software, as the basis for verification activities, Dr. Buth asserted that criticality assessment is the first step in determining quality assurance activities. Using hazard and operability analysis (HAZOP) for high-level effects of the system and software fault tree analysis (SW) FTA and FMEA for the identification of causes for worst-case scenarios, the project is able to identify functions, components, and interfaces between components, which contribute to critical situations. Decisions can then be made as to the most efficient use of QA activities, (code inspections, static and dynamic analysis, coverage tools, reviews, checklists). Formal methods were touched upon as an alternative; however, this was viewed as of indeterminate or questionable value.

A textual example of a fault tree analysis for a controller of belt speed for incoming and outgoing product was described. One way of looking at this situation is to view the leaves on the fault tree as control elements for the FMEA. Error situations,

then, in FMEA are the leaves of the fault tree. FTA provides the severity category for the FMEA. Positive experiences have been reported by NASA using this strategy, on software reuse and in the safety analysis for tilt control of trains in Europe.

In summary, the benefits of this strategy are claimed to be the ability to use the combined results of FTA and FMEA to analyze component interaction, dependencies between data related failures, select test cases, and run robustness tests thereby making the most effective and efficient use of quality assurance activities.

Looking With Software Eyes at Digital Devices

Ilja Waigl Senior Qualification Engineer Airbus Deutschland GmbH

This presentation is one of the better discussions that explain the similarities and differences between application specific integrated circuits (ASICs) and programmable logic devices, (PLDs), and microprocessor controlled systems, and how they are designed, manufactured, and verified.

An ASIC is an integrated circuit in which logic states are programmed into factory masked ROM. They perform a usually repetitive function such as encoder/decoder for ARINC 429.

Whereas ASICs have a fixed configuration, PLDs can be re-programmed to modify the implemented function. The function of the PLD consists of a simple programmable logic array and a set of input and output connectors. PLD are used typically for smaller and less complex application than ASICs.

The basic difference between these devices and microprocessors can be described as "computing in space" vs. "computing in time." A microprocessor contains an instruction set that must be executed in time, step by step. Accordingly a microprocessor executes in time. An ASIC/PLD uses information stored in a spatial array and evaluates input signals and provides appropriate outputs based on those inputs. Accordingly, an ASIC/PLD executes in space.

More specifically the differences are:

Microprocessors:

- Operate in time and value
- Are equipped with an instruction set
- Have functionality defined in software
- Use instructions executed step by step
- Use high-level languages

Digital devices:

- Use basic Boolean operators
- Operate as finite state machines
- Function with logical expressions
- Scenarios shift stepwise through logic
- Functionality is spatially arranged.
- Hardware language is used.

Software testing for microprocessors is conducted on compiled code at 3 levels, according to RTCA/DO178B: hardware-software integration testing, software integration testing, and low level testing. Depending on the test scope, one or more instruction sets are linked and tested.

Digital devices are constructed first by creating an abstract model using a hardware description language. Once the model is built it undergoes testing via simulation of the hardware functions.

The next step transfers the software model to the register transfer level (RTL), which considers target hardware specific

properties. Next a synthesis in logic and layout produces the final netlist that is implemented in hardware.

For certification purposes, the authorities require extensive testing of software per RTCA/DO-178B. As a result of the increased complexities of ASICS and PLDS, EUROCAE WG-46 and RTCA SC-180 were formed and produced RTCA/DO-254 for the development and certification of digital devices.

Authors note: Prior to the issuance of D0-254, there were spirited debates between software and hardware developers and the regulatory authorities on whether digital devices really were considered software. Although there are many similarities in the process terminology and documentation suite, ultimately digital devices are not considered software. Nonetheless they need very similar process discipline and assurance.

Experiences From Agile Development Projects

Martin Lippert, University of Hamburg (D)

Martin Lippert is a research assistant at the University of Hamburg, a professional software architect and consultant at IT Workplace Solutions. His current research work includes framework design, tool support for framework specialization, refactoring, JAVA, and extreme programming. He is the author of *Extreme Programming Examined*, *Software Quality and Software Testing in Internet Times* and co-author of *Extreme Programming in Action: Practical Experiences from Real-World Projects*.

Lippert began with an overview of what he called a “manifesto” for agile methods. He reported a new set of values,

Agile methods		Traditional
Individuals	vs.	processes & tools
Working sw	vs.	detailed documents
Customer collaboration	vs.	contract negotiations
Responding to change	vs.	following a plan

The main ideas of agile methods are:

- Adapt the process to the environment.
- Create as much business value as soon as possible.
- Deliver early, deliver often.
- Make the process lightweight.
- Get rid of unnecessary documents.

He espouses “Test-First Design, and Test-First Programming.” Test-First Design includes defining the interfaces of the class, specifying the boundary conditions, improving modularity, minimizing dependencies, and simplifying the design. Test-First Programming involves changes driven by test failures, writing just enough code to see the test fail, then writing just enough code to see the test pass, then refactoring the code into its simplest form.

Lippert concedes agile development within short cycles becomes difficult in the presence of databases. This may require the integration of the code with the changed database schema which is expensive. His solution for this is to ensure that database know-how has been integrated into the teams. Each update from the teams contains the needed database scripts as well as the changed code.

Lippert concluded by promoting books on extreme programming including his own, and gave a short bio on his company. As with Beck’s talk, no objective industry data on the effectiveness of agile methods or extreme programming were supplied.

For further information on any of these presentations and for a complete set of proceedings on CD, please contact me at 425-717-7038 or by e-mail at michael.p.kress@boeing.com.

Michael Kress is an associate technical fellow for the Boeing Commercial Airplane Group in Seattle. He is a Senior member of ASQ and currently is the chair of ASQ’s Software Division.

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
 iqps@aol.com • www.iqps.net

CERTIFICATION

DOUGLAS B. HAMILTON

A CSQE Exam Review Committee meeting was held September 12 and 13, 2003, in Milwaukee. I would like to thank the following volunteers for making the session a big success: Karen Bishop-Stone, Nancy Casteel, Terry Deupree, Mary Frisch, Alex Hilgendorf, Evelyn Richardson, Deana Seigler, Robert K. Smith, Bob Stoddard, Rocky Thurston, and Matt Wilson. In addition, a special thanks to Alex Hilgendorf for recruiting the volunteers and chairing the session.

The volunteers are what make the exam a success. The meetings are fun, challenging, and you learn something every time. If you are interested in volunteering, you must be a CSQE. Please mail your resume to Mary Martin at mmartin@asq.org for consideration for future workshops.

Region 1 Eric Patel

Preparation for activities this fall has started. The QAI Boston Federation Chapter, the Boston Quality Assurance Association (BOSQAA), has expressed interest in working with Region 1 to promote software quality in and around the Boston area. One of BOSQAA's goals is to host monthly meetings where QA professionals can come together for networking and learning. Instead of the typical format of a speaker, BOSQAA meetings will be more of an open discussion (e.g. "birds of a feather") whereby a topic will be discussed among the meeting participants. Those interested in helping out or joining BOSQAA can contact me at epatel@rapidsqa.com.

John Pustaver and I will kick off the 10th season of the Software Quality Group of New England (SQGNE) meetings Wednesday, September 10 from 6 to 8 pm at Sun Microsystems in Burlington, MA. For more information or to speak at an upcoming meeting, visit www.swquality.com/users/pustaver/bcs.htm or contact me at epatel@rapidsqa.com.

Region 2 Jean Burns

Region 2 is looking for two deputy councilors to work with Jean Burns during 2003 and 2004. A deputy councilor assists the regional councilor, facilitates communications in the region, and in general assures that software quality programs and information are available in your local area. Those of you in the Corning-Elmira, Buffalo-Rochester, Syracuse, and Albany areas are most welcome to contact Jean Burns at 607-779-7868 or burns@uic.com to discuss how you can work with her to better serve the region.

Jean Burns is available to attend your local sections to speak about the Software Quality Division, the CSQE exam, or other topics of interest in software quality.

The Rochester, NY, Section is planning their quality conference for March 2004.

If you have any upcoming events you want discussed in the Software Division newsletter please contact Jean.

Region 4 Chris Fitzgibbon

September promises to be the start of

a very busy season for events that appeal to the software quality practitioner in Region 4. There is an excellent choice of conferences and, after a brief summer break, many local ASQ sections and software quality associations are set to resume their monthly meeting schedule. 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 or 613-563-9000. It would be a pleasure to hear from you.

Conferences

There is still time to register for the 13th International Conference on Software Quality (ICSQ) in Dallas on October 6 to 9, 2003. This conference promises to be one of the best software quality events of the year. To get more information or to register, check out the conference Web site at www.icsq.org.

On October 2, 2003, the ASQ Toronto Section will hold its annual single day conference. Considerable effort has gone into making this event appealing to the software quality practitioner. The Web site is: www.torontoqualityforum.com. Also in Toronto is the Quality Assurance Institute (QAI) International Quality Conference 2003. The conference is on October 1-3, 2003; additional information is available at www.dkl.com/conference/main.html.

One of the largest annual quality-related conferences is the ASQ's Annual Quality Congress. We are very fortunate to be hosting the 58th AQC in Toronto May 24-26, 2004. Although the attendance at the conference will be much less than the recent record-setting Rolling Stones & Friends rock concert, the event will be equally memorable for the quality practitioner. There will be a software track and many other excellent speakers. You won't want to miss this one! More information is available at the AQC Web site: <http://aqc.asq.org>.

Calgary, Edmonton, and Vancouver

The Calgary IEEE/ASQ Discussion Group for Software Quality has several events already scheduled for this fall. Topics include "Selling QA to Management," "Why Automate Software

Testing," and "Agile/SCRUM for Software Acquisition." 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.

Toronto and Southern Ontario

The Toronto Association of Systems and Software Quality (TASSQ) holds dinner meetings the last Tuesday of each month at the Sheraton Centre Toronto Hotel across from New City Hall. Some recent topics include Test Automation and Bug Tracking Solutions. Upcoming events are available at www.tassq.org. The fall sessions of the Toronto SPIN and the ASQ Toronto Section meetings are also being planned. Check the Toronto SPIN's Web site for future planned SPIN events: www.torontospin.com. The Toronto Section's Web site is www.asq-toronto.com.

Ottawa and Montreal

The ASQ Ottawa Valley Section has established a "Software Improvement Focus Team" with a mandate to promote software quality throughout the section. The group has developed a strategic plan and will co-ordinate a CSQE study group this fall. If you are interested in participating in the study group, contact me (chris@orioncanada.com) and I will forward additional information.

The Ottawa Software Quality Association is looking for suggestions for topics and speakers. They welcome your feedback and inquiries at suggestions@osqa.org. The September session will include several roundtable discussions on software quality challenges and solutions. The calendar of events for the Ottawa SPIN (www.spin.org) and Montreal SPIN (www.spin-montreal.org) should appear on their Web sites soon.

Region 6 Tom Gilchrist

This summer, SASQAG (the Seattle Area Software Quality Assurance Group) held another of its "S99 Training Days" on Friday, June 13, 2003. The workshop, "Use Cases for Maximizing Project Success," was presented by Carol Dekkers. We had our largest turnout to

date with 60 people attending with some who were turned away because of the venue. These events are designed to be full-day workshops and/or tutorials with food service at a reasonable price (\$99). For information on this event or upcoming events, visit www.sasqag.org/99days.

The Pacific Northwest Quality Conference organization is presenting their annual quality conference October 13-15 at the Oregon Convention Center in Portland, OR.

Keynote speakers this year will be Cem Kaner (How Many Light Bulbs Does It Take to Change a Tester?) and Brian Marick (Agile Testing: A Year in Review). There will be technical papers and other invited speakers as well as workshops and exhibits.

For more information, visit <http://www.pnsc.org>.

On the third Thursday of every month

(except December), SASQAG holds monthly 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 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.

Tom Gilchrist, Region 6 ASQ Software Division, tomg@tomgtomg.com.

Region 8 Michael Kiefel

On June 14, 2003, I attended the Region 8 Leadership Training Session

conducted by Region 8 director Fletcher Birmingham. There were 37 attendees representing 8 sections out of the 12 in Region 8. This was the first year that divisional regional councilors were encouraged to attend this training. Four divisions were represented by regional councilors and included the Software, Health Care, Quality Management, and Measurement Quality Divisions. Each regional councilor addressed the attendees with objectives, upcoming events, and contact information. Due to the need by the health care industry to move to electronic records, the Region 8 councilor for the Health Care Division expressed a desire to connect with members of the Software Division for insight on the development process and quality issues. Any Software Division member willing to become involved in discussions with the Health Care Division on

ANSWERS TO THE SOFTWARE QUALITY ENGINEERING QUIZ

BY LINDA WESTFALL

- 1. Answer A is correct.** This is the definition of a standard. A procedure is a written description of a course of action to perform a given task. A process is a sequence of steps performed for a given purpose. A guideline is a recommended but not required approach. **CSQE Body of Knowledge Area: I.B**
- 2. Answer D is correct.** External failure costs are all the costs of quality that involve handling and correcting a failure that has occurred once the software has been made available to the customer. The cost of maintaining a Web site where customers can obtain service pack downloads would be counted as an external failure cost. **CSQE Body of Knowledge Area: II.B.3**
- 3. Answer B is correct.** A capture and playback tool can be used to do test automation by capturing the inputs the tester enters during the execution of a test case or test scenario. Those sequences can then be "played back" to repeat the testing sequence. **CSQE Body of Knowledge Area: III.D.4**
- 4. Answer D is correct.** Process is not one of the three major elements that must be balanced to have a successful project. The cost element represents the cost of all the resources that go into the project. The schedule element represents the calendar time it takes to complete the project. The product element represents both the quality and the quantity (amount of functionality) of output from the project. The larger the resulting product, the longer it typically takes and the more it costs. **CSQE Body of Knowledge Area: IV.A.1**
- 5. Answer A is correct.** The Goal/Question/Metric Paradigm looks at the goals of the metric customer, asks questions about the types of information needed to determine whether we are moving toward those goals or have achieved those goals and then selects metrics to provide the information needed to help answer those questions. This method is used to ensure that we are selecting metrics that align with the goals of the metric customer. **CSQE Body of Knowledge Area: V.A.3**
- 6. Answer B is correct.** Formal inspections are peer reviews and are therefore NOT attended by project management (with the exception of when a project management work product is being inspected). Attendees of a formal inspection are assigned specific roles and responsibilities including moderator, author, reader, scribe, and inspector. Formal inspections focus exclusively on detecting defects in work products that the author considered complete. Therefore, inspections do NOT provide engineering analysis into the creation of the work product. **CSQE Body of Knowledge Area: VI.B.1**
- 7. Answer C is correct.** The functional baseline is established when the system requirements are approved. The allocated baseline is established when the system requirements are allocated to the software and the software requirements are approved. A developmental baseline is established when the software design specification has been approved. A product baseline is established when the software has been approved for distribution to the field. **CSQE Body of Knowledge Area: VII.B.2**

FROM THE REGIONS

CONTINUED

this subject may contact me at michael.kiefel@abbott.com for further information.

As a reminder, the 4th Annual Michigan Quality Conference will be held October 23, 2003, in Canton, MI. This conference includes an all-day software related track. Anyone interested in attending this conference from central and Southeast Ohio and would like to carpool may contact me at michael.kiefel@abbott.com.

Region 10 Nancy Poma

Michigan Quality 2003—Here is what you have been waiting for—the software topics and speakers for our 4th annual one-day conference, to be held October 23, 2003, at the Yazaki Learning Center in Canton, MI.

- Establishing Meaningful Metrics for Performance Management—Elaine Rusnak
- Tackling the Software Testing Maze (from chapter 1 of her book)—Louise Tamres
- Defect Prevention Techniques—In Automotive Embedded Systems—Ed Neubecker
- Overcoming Roadblocks to Process Improvement - Blythe Williams
- Effective Project Planning Without Microsoft Project - James Goebel

I want to recognize the contributions of Debra Beaman and Michael St. Peter who have helped plan the software track and the overall conference both this year and last year. This year's event will also include tracks on automotive and customer/supplier quality. Breakfast, lunch, and handouts are provided for all three tracks. If you need more information, please contact me at nmpoma@comcast.net.

Local Meetings of Interest

The next GL-SPIN meeting is on October 9 is at Oakland University. For more information on upcoming events, check out the Web site at www.gl-spin.org.

The Greater Detroit and Ann Arbor Sections and the Automotive Division will host a joint meeting at Madonna University September 18, 2003. Chuck Fellows will present on “Demystifying

Six Sigma.” For more information on upcoming events, check out the Web site at www.asqdetroit.org.

Region 12 Irv Segal

Exciting things are in the works for Region 12!

SysGen, Inc. (my employer) has dedicated resources for planning and hosting several events throughout the year on an ongoing basis in conjunction with ASQ Software Division. The focus of these events will be to promote new ideas, techniques, and tools for addressing software quality assurance in real world situations. Textbooks, standards, trade books, articles, etc., provide us with more than enough great SQA ideas to make our heads spin. But how do we implement these good software quality practices when the budget manpower and time are rarely available to do so?

These events will focus on how you and I can actually implement SQA practices with whatever limitations we have to work with.

Our first event is already in the preliminary stages. We're planning a 1-2 day conference and vendor exposition focusing on tools and methods that can streamline and reduce the SQA workload given today's tight budgets and project timelines. This will be held in the Chicago area either late 2003 or early 2004.

We've already started lining up speakers and vendors. I encourage members to contact me with topics or vendor solutions they'd like to see addressed or presented. What are the problems you encounter on a day to day basis? Is there a product you've been curious about but haven't had the time to investigate? Or perhaps you're wondering (or wishing) there were a product out there to solve a particular problem you're having. Maybe you'd like to share how you get things done at your SQA shop.

Contact me for any of these or just to share thoughts and ideas for Region 12 programming at asq@sysgeninc.com.

Region 13 Grandville Jones

Items for the upcoming year include working with Robin Dudash of the Education Committee to put together a directory of CSQE resources. We hope to publish the directory in the newsletter annually with updated information each year. We are also investigating its possible link to the Quality Resources section on the ASQ Web site. If anyone in Region 13 would like to volunteer to help on the

project call or e-mail me.

The Denver Section SW Committee is planning to sponsor a speaker at one of the monthly membership meetings this fall. Each meeting has a different theme with one being high-tech process improvement.

Feel free to contact me if you have any questions about Region 13 and software. If I can't answer your questions I'm sure someone on the board can.

Region 14 Bill Trest

Made contact with ASQ Section 1416 (Greater Fort Worth) to offer councilor support/information regarding CSQE certification. My hope is to make further contact with ASQ sections throughout the region.

Region 15 Mark Neal

The Region 15 Planning Team consists of the following volunteers: Mark Neal (Region 15 councilor), Carol Dekkers, Scott Duncan, Theresa Hunt, Michael Kirchner, and Martha Purser. In this past quarter, the Planning Team has met a number of times to identify the types of activities that members in our region would benefit from. We have developed a survey and sent it to our members to gauge their interest in the following opportunities:

- Providing help in preparing for the Certified Software Quality Engineer (CSQE) exam. This help may include identifying local study groups of members, summarizing training materials and classes available, and/or identifying CSQE mentors to ask questions.
- Identifying or establishing periodic Software Quality “Roundtable” discussions in the local areas of the region.
- Attending a one day mini-conference within the region.
- Other comments or needs of the members.

We will be reporting the results of the survey and the plans established by the planning team by the end of August.

We have also started more frequent email communications to our members. Please note that only two-thirds of the region members have their ASQ e-mail preferences set to receive e-mail from the Software Division and the Region 15 Planning Team. If members wish to receive these communications, their e-mail preferences must be set accordingly in their online member profile.

RECAP OF AQC 2003-KANSAS CITY, MO MAY 19-21, 2003

SOFTWARE DIVISION TRACK: YOUR WORLD DEPENDS ON QUALITY SOFTWARE

BY CAROL DEKKERS,
AQC SOFTWARE DIVISION TRACK CHAIR

We're back from the city of BBQ's and baseball! Once again the Software Division presence at the Annual Quality Congress (AQC) was impressive. Our seven sessions over the three days of the congress presented an impressive lineup for attendees from a variety of quality specialties. I heard many great comments about both the speakers and the topics – in fact, ASQ headquarters was so impressed with our division track that we've been invited to host a software quality track at the upcoming next two congresses: 2004 in Toronto, Canada, and 2005 in Seattle, WA. This is all the more prestigious when you consider that less than a handful of ASQ divisions are bestowed this honor each year. 2004 will be our third consecutive year to have an AQC track dedicated to software quality and it is with thanks to many within our division that this is possible.

Software Division Track Recap

If you missed the presentations this year, you might be interested to know that we are currently in the process of producing CD copies of the Software Division track "Your World Depends on Quality Software" presentations complete with the slides (in pdf format), and, for four of the sessions, an audiotape recording of the session. It's almost as good as being there! Watch for news about release date and pricing of the Software Division "Your World Depends on Quality Software" CD in the next newsletter and on the Software Division Web site!

To recap the Software Division presentations featured at AQC in 2003: (audiotaped presentations denoted by an asterisk)

- **Certification of Software Professionals: The Benefits of the Certified Software Quality Engineer (CSQE) Designation to You**, featuring leading experts: Rufus Turpin of Carpe Diem Informatics, Inc.; Doug Hamilton of Accenture; and Taz Daughtrey, editor of the *Software Quality Professional* journal.
- **Software Validation in a Regulated Manufacturing Environment**, (joint session with the Biomedical Division) featuring David Chevin of DHC Associates and Dan Olivier of Certified Software Solutions, Inc.
- **HIPAA – Where Health Care and Software Quality Meet**, featuring Peggy Bowling of LEO Management & Associates, Inc., and Ron Berglund.
- **Software Acquisition and Supplier Management**, featuring Linda Westfall and Theresa Hunt of the Westfall Team.
- **Maximizing Your Productivity on a Software Project: Use Cases for Users**, featuring Carol Dekkers of Quality Plus Technologies, Inc.

- **Measuring Quality – Best Practice Approaches**, featuring Thomas Ehrhorn of Framatome ANP DE & S, and Taz Daughtrey of James Madison University and the *SQP* journal.
- **Planes, Trains, and Automobiles—What's the Connection to Software Quality?** Featuring John Pawlicki of EDS (US).

The Software Division "Your World Depends on Quality Software" CD will be available soon. To reserve your copy, please send an e-mail to Carol Dekkers (Dekkers@qualityplustech.com) with the subject Software Division CD.

Thank you's

Thank you to the Software Division Council and especially to David Walker, Sue Carroll, Rufus Turpin, Hank Sobah, Jayesh Dalal, Theresa Hunt, Tim Surratt, Mike Kress, Joel Glazer, Taz Daughtrey, Doug Hamilton, Linda Westfall, Tom Ehrhorn, Terry Deupree, Hillel Glazer, and everyone else who moderated sessions, spoke, or generally assisted to make AQC 2003 a success for our division. It is invigorating to note that many of these dedicated volunteers have signed up for the 2004 AQC committee – it will be my pleasure to work with all of you again.

AQC 2004

Plans are already under way to fill out our AQC track for 2004 with details to be announced in upcoming newsletters. If you are planning to be in Toronto for AQC in May 2004, or if you are interested in speaking or moderating sessions at the conference, please e-mail me (Dekkers@qualityplustech.com) to volunteer. We'll find a job that meets your skills and your interest!

Until next newsletter, I hope your software (quality) endures.

Carol Dekkers can be reached by e-mail at Dekkers@qualityplustech.com.

SPOTLIGHT

MEASURING THE "LOGICAL" OR "FUNCTIONAL" SIZE OF SOFTWARE PROJECTS AND SOFTWARE APPLICATION

BY CAROL A. DEKKERS,
U.S. DELEGATE TO ISO/IEC SC 7, IFPUG
REPRESENTATIVE TO ISO/IEC SC 7

Function points in a nutshell is "square meters for software." In other words, user requirements are the floor plan and function points is the size of that floor plan. ISO/IEC and the International Function Point Users Group (IFPUG) are now publishing ISO/IEC 20926: IFPUG Function Point Counting Practices 4.1 unadjusted as an international standard.

This standard reflects a collaborative effort of industry and formal standardization within IFPUG and ISO/IEC to bring what was already the de facto world standard for functional size measurement to the ISO stage. The initiative has included a wide spectrum of participants from industry, academia, government, and the software development community. To facilitate this event, IFPUG followed the formal ISO/IEC JTC1 Publicly Available Specification (PAS) process which was set up to bring standards already established and in widespread industry usage to ISO publication.

The publication of ISO/IEC 20926 represents a landmark in both ISO/IEC history and in the recognition of functional size measurement in the software industry because it brings a solid functional sizing methodology to the ISO marketplace. Long before Alan Albrecht of IBM first introduced the basic concepts of what is now known as "functional size measurement" in 1979, the software industry has struggled with the problem of quantifying project or base application (installed software) size when developing or maintaining software. While other established engineering-based disciplines such as building construction and manufacturing are able to objectively assess project size when estimating or comparing projects, the software industry has long been challenged to find suitable measures for quantifying the various aspects of software.

Functional size quantifies the functionality dimension of software, and is a component along with quality and technical requirement measures on which one can base project size. Susan Burgess, corporate vice president of software and systems quality of Veridian Inc. and a key contributor to ISO/IEC 15288 (System Life Cycle Processes), ISO/IEC 12207, and others, heralded the announcement of ISO/IEC 20926: "The publication of ISO/IEC 20926 (2003) at last recognizes the importance of functional size measurement to the IT industry. IFPUG 4.1

unadjusted has assumed its rightful place alongside key ISO/IEC measurement standards including ISO/IEC 15939, 14598 and the 9126 suite of standards. ISO/IEC 20926 was sorely needed to satisfy measurement concepts of ISO/IEC 15504, and also to facilitate progress indicators for ISO/IEC 12207, among others."

The phrase functional size measurement is defined in ISO/IEC 14143-1:1998 Functional Size Measurement – Definition of concepts as "the process of measuring functional size." "Functional size" is, in turn defined as "a size of the software derived by quantifying the functional user requirements." IFPUG function points are the most widely accepted units of functional size measurement in use today as demonstrated by their usage in myriad countries throughout the world including: Argentina, Australia, Austria, Barbados, Brazil, Canada, China, Denmark, Finland, France, Germany, Holland, India, Ireland, Italy, Japan, Korea, Mexico, New Zealand, Portugal, Saudi Arabia, South Africa, Spain, Sweden, the United Kingdom, the United States, Venezuela, and others. In fact, any country where its software industry has embraced functional size measurement is likely to be using IFPUG function points or a variant thereof. Mauricio Aguiar, principal of ti Metricas, a leading measurement consultancy in Brazil, heralded the news about ISO/IEC 20926 by stating: "The publication of ISO/IEC 20926 is an important step to foster the adoption of IFPUG function points by governments around the world. This has happened to a large extent in Brazil, Australia, and Korea. It will certainly happen in many more countries in the future."

IFPUG Function Points

IFPUG function points measure the "logical" or "functional" size of software projects and software applications based on the functional user requirements (FUR). FUR represent the processes and procedures performed by the software – in other words, "WHAT" the software will do – and can be sized by IFPUG function points. Functional size is independent of how the software will operate (i.e., the quality and performance requirements) and also of how it will be developed. ISO/IEC 20926 applies to all domains or types of software without exclusion, and has been successfully applied to measure functional size for software ranging from missile defense systems to financial reporting software. Since 1986, the International Function Point Users Group, a not-for-profit organization headquartered in the USA with members worldwide, has maintained the IFPUG function point rules through a rigorous change management process and a balanced committee of measurement practitioners representing a variety of industries and software development organizations. Capers Jones cited in 1999 that "IFPUG counting rules are used by at least twice as many people as all other counting methods put together." The International Software Benchmarking Standards Group (ISBSG) database Version 7 (2002), demonstrates an even higher margin of usage—an overwhelming number of its projects are sized in IFPUG function points.

What is Functional Size?

A couple of definitions can help with the understanding of how ISO/IEC 20926 measures the functional size of software. The word "user" in the context of functional size measurement means any person, thing, other software, department—anything outside the boundary of the application, that has a need to interface (i.e., interact) with the software. This definition of user is

analogous to “actor” in object oriented development, and is an expansion of what the Information Technology community might typically describe as a “user” to include other software

“ It is a credit to the ISO community, and the IT world, that the PAS process exists— it became the conduit to bring the publication of IFPUG Function Points to fruition as ISO/IEC 20926.”

applications and things besides physical persons.

The software development community urgently required the IFPUG 4.1 Unadjusted (ISO/IEC 20926) to facilitate software sizing for project estimating, productivity and quality analysis, and to support other project and portfolio decision making. Capers Jones, chief scientist of Artemis, Inc., and a prolific author of software engineering, estimating and measurement books, stated: “Function points provide the only metric capable of measuring the economic productivity of software products... no other metric can handle all of the non-coding tasks associated with software.”

What is the ISO/IEC JTC 1 PAS process and what does the publication of ISO/IEC 20926 mean to my organization?

John Hippen, U.S. technical expert to ISO/IEC JTC 1/SC 7, a key contributor to ISO/IEC 15504 (SPICE), and the former Australian head of delegation to ISO/IEC JTC 1/SC 7, summed up the IFPUG experience in the PAS process by stating: “The PAS process facilitated the realization of a dream first established in 1993 to publish IFPUG function points as an ISO/IEC standard. I was part of the core ISO/IEC team that created the functional size measurement project, and at that time its goal was to publish the IFPUG Function Point standard already in widespread usage—as an ISO standard. Along the way, other goals such as framework and concept standards took precedence on the project. It is a credit to the ISO community, and the IT world, that the PAS process exists – it became the conduit to bring the publication of IFPUG function points to fruition as ISO/IEC 20926.”

The ISO/IEC JTC1 publicly available specification (PAS) process itself is a two-step process consisting of two separate ISO/IEC ballots. The PAS process facilitates and “fast tracks” existing industry standards already in widespread usage to become international standards. Unlike the regular ISO/IEC standard development process where countries participate in the development of new international standards, the PAS process brings to the ISO/IEC marketplace those specifications already established and accepted by industry. In addition, only those specifications supported by a stable and established organization can be considered for PAS processing. When IFPUG completed the first step of becoming an ISO PAS submitter in 1999, the whole process appeared to be daunting at first glance, however, with the cooperative and eager assistance of both the JTC 1 secretariat and the ITTF office in Geneva, the process was made easier to navigate and understand. The first step to becoming an ISO/IEC standard requires that the organization that owns the PAS be formally recognized and accepted as a bonafide PAS submitter organization by a voting majority of member countries to

ISO/IEC JTC 1/SC 7. This involves filling out a submitter application and a formal explanation of longevity, cooperation with ISO standards work, and proof of stability by the candidate submitter organization. Once an organization has been accepted as a PAS submitter organization, it must prepare and format the specification for the JTC1 ballot of the standard itself. This is a six-month ballot that again must be approved by the majority of JTC1 voting countries. IFPUG passed this ballot with a majority vote in late 2000 and has since been preparing for ISO publication through resolution of dissenting countries comments. In the same way that the ISO/IEC standard development process seeks to ensure as much consensus as possible in the development and publication of ISO standards, the PAS process also seeks to achieve unanimity (in so far as is possible) with standards that progress through PAS processing. This means that standards already in widespread industry usage that your company or your country may have embraced could become ISO standards by utilizing the PAS process.

James Moore, former chair of the U.S. Delegation to ISO/IEC JTC 1’s SC 7 stated: “When operating at its best, the ISO/IEC PAS process serves to take existing technically excellent products, remove cultural biases towards the industrial practices of the originating country, and broaden their exposure and usage on the global stage. In this case, the IFPUG specification already had broad usage around the world. The minor changes demanded by the ISO/IEC adoption process reaffirm the role of the IFPUG specification as the wellspring of functional size measurement methods.”

After several years of hard work, it is an outstanding achievement to bring IFPUG 4.1 unadjusted to the world stage as ISO/IEC 20926—particularly as it is the leading functional size measurement method in use today, and supported by hundreds of organizations that belong to the International Function Point Users Group.

For further information on the IFPUG 4.1 unadjusted standard (ISO/IEC 20926) or functional size measurement as an emerging software engineering practice, visit the IFPUG Web site at www.ifpug.org. To contact the author of this article, send an e-mail to Dekkers@qualityplustech.com.

“ When operating at its best, the ISO/IEC PAS process serves to take existing technically excellent products, remove cultural biases toward the industrial practices of the originating country, and broaden their exposure and usage on the global stage.”

How Does the ISO/IEC 20926 Measure Software Functional Size?

ISO/IEC 20926 evaluates the software to be sized by breaking down its functional user requirements into five types of base logical components (BLC):

Internal logical files (ILF), are the persistent logical groupings of data (entities) that are maintained through one or more standardized elementary processes of the software. In plain English, the ILFs of a software application are the maintained data group-

(cont. on p. 22)

ings/objects/entities that are maintained through a standardized process or procedure. (Historically, maintained means one or more of the Create or Update or Delete of a traditional CRUD matrix.) Once an entity is classified as an ILF, there is a weighting step that objectively rates each ILF as low, average, or high and subsequently assigns a function point “weight” of 7 FP for low, 10 FP for average, and 15 FP for high category ILFs.

External interface files (EIF) are the persistent logical groupings of data that exist within another software’s boundary and are reference only (but not maintained) by the software being sized. This means that an EIF must be (or would be) an ILF within another software application. Once an entity is classified as an EIF, there is a weighting step that objectively rates each ILF as low, average, or high and subsequently assigns a function point “weight” of 5 FP for low, 7 FP for average, and 10 FP for high category EIFs.

External inputs (EI) are the elementary functional processes whose primary intent is to either maintain one or more ILFs OR control the behavior of the application (i.e., trigger an event to occur that might otherwise not occur). Each unique, standalone, self-contained elementary process with the primary intent as just described counts as an EI and is rated through the IFPUG method into low, average, or high and scores 3, 4, or 6 FP, respectively.

External outputs (EO) are the elementary functional processes whose primary intent is to present data to a user and include one or more of the following criteria within the process:

calculations, derived data, updating of an ILF, modification of application behavior. Each unique, stand-alone, self-contained elementary process with the primary intent as just described counts as an EO and is rated through the IFPUG method into low, average, or high and scores 4, 5, or 7 FP, respectively.

External queries (EQ) which are the elementary functional processes whose primary intent is to present data to a user from one or more ILFs/EIFs and the process cannot include any of the following criteria: calculations, derived data, updating of an ILF, modification of application behavior. Each unique, stand-alone, self-contained elementary process with the primary intent as just described counts as an EQ and is rated through the IFPUG method into low, average, or high and scores 3, 4 or 6 FP, respectively.

Once the base logical components (BLCs in the functional size measurement terminology) have been identified, evaluated and scored, the total function points are summed and the resultant integer number represents the functional size of the software.

Carol A. Dekkers has been a key Category C liaison representative for IFPUG (www.ifpug.org), a member of the IFPUG ISO task force, and a U.S. delegate to ISO/IEC’s JTC 1/SC 7/WG 12 since 1994. She is a past president of IFPUG, and is also active in the American Society for Quality (ASQ), and the Project Management Institute (PMI). Professionally, Dekkers is president of Quality Plus Technologies, Inc., where she is in demand as a consultant, instructor, advisor, author, and mentor for companies truly interested in achieving breakthrough process improvement through measurement. Visit the IFPUG Web site at www.ifpug.org for more information about function point analysis. Dekkers can be contacted at dekkers@qualityplustech.com.

OFFICERS

Michael P. Kress, Chair
The Boeing Company
425-266-0545
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
LFI Group Inc.
613-299-2770
joneslf@magma.ca

Taz Daughtrey, Liaison Chair,
Journal Editor
804-237-2723
sqpedit@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—Bill Folsom
Defense Contracts Mgmt. Agency
203-385-4339
wtfolsom@dcmde.dcmil

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

Region 5—Joel Glazer
Northrop Grumman ES
410-765-4567
jglazer@northropgrumman.com

Region 6—Tom Gilchrist
The Boeing Company
425-965-6051
tomg@tomgtomg.com

Region 7—OPEN

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

Region 9—Rob Price
Escient Convergence Corp.
317-616-6557
rprice@escient.com

Region 10—Nancy Poma
EDS
248-262-7820
nmpoma@comcast.net

Region 11—Greg. B. Jones
Bank of America
704-683-2239
greg.b.jones@bankofamerica.com

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

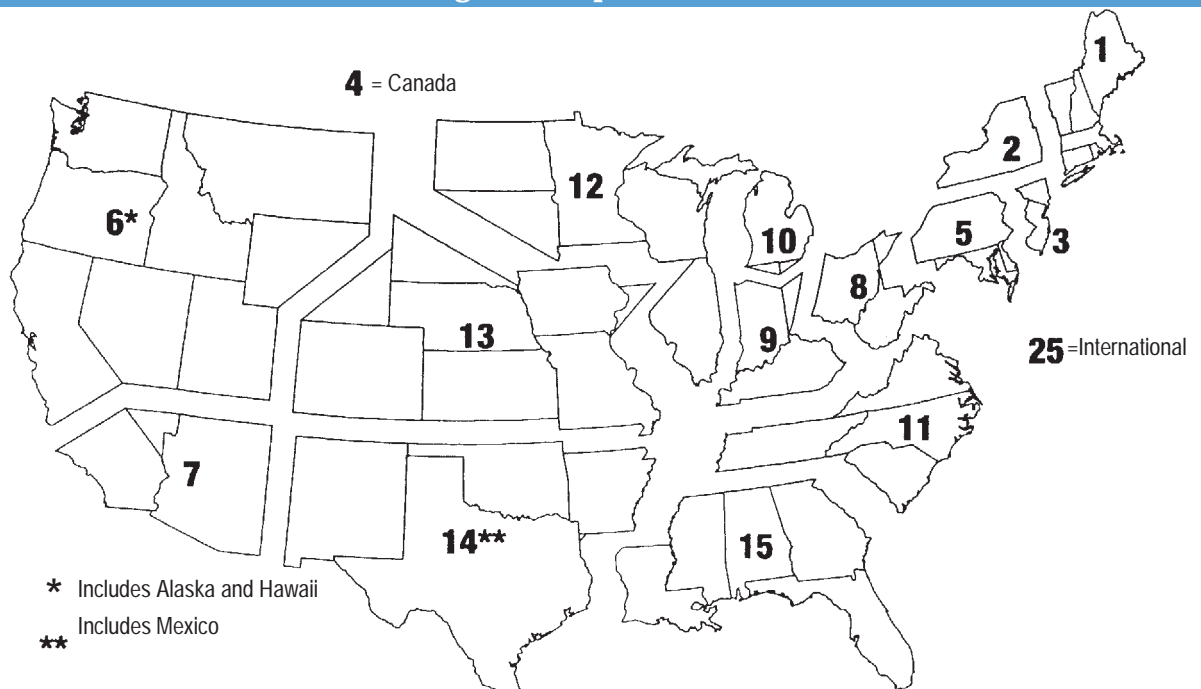
Region 13—Granville Jones
JVJ Enterprises
303-969-0228
granville.jones@att.net

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
DECEMBER 1, 2003.**

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
Doug Hamilton, 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

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://sfp.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