

Software Quality Matters

Volume 5, Number 4, Winter 1997

Articles in this issue:

Software Industry Must Follow Same Quality Principles as Other Technological Industries by Watts Humphrey

SQI to Offer Personal Software Process Training for Software Developers in '98

2nd Annual Telecom Conference Slated for Next February in Austin

2nd Annual Telecom Conference Slated for Next February in Austin

NASA's Shuttle Software: Lives Depend on Its Quality by Paul Ehrler

How Software Development Is Like the Oil in Your Car by Celeste Armstrong

Austin to Host IEEE International Multimedia Conference this Summer

Successful President/CEO Peer Group Marks Good Start to ASC Initiative by Karen Jonson

Software Industry Must Follow Same Quality Principles as Other Technological Industries

by Watts Humphrey

(Editor's note: A slightly expanded version of this article was submitted to the National Conference of Commissioners of Uniform State Laws. This body is now drafting a proposed new article 2B, which deals with the information industries. If adopted, article 2B will govern the way software contracts are interpreted in the United States.)

The current state of software quality is best described by the commonly held view that "It is impossible to remove all defects from software products." This statement, while widely accepted, is incorrect. Many defect-free software products have been developed.

A more accurate way to state the current situation is that "It is generally impossible to prove that all defects have been removed from a software product."

There are several reasons for this. Many software products are extraordinarily complex. It is not possible to exhaustively test these products. Thus, if one relies exclusively on testing, it is truly impossible to demonstrate that a software product is defect free.

Industrial Quality Principles

While the software industry has had a troublesome history, there is no reason for it to follow different quality principles from those used in other technologically intense industries.

Modern commercial aircraft, for example, are very complex hardware and software systems. One cannot definitively prove that all design and manufacturing defects have been removed from a large aircraft. This, however, does not prevent aircraft manufacturers from warranting their products or from taking full responsibility for product quality. Many other industries produce high-quality products and take full responsibility for any resulting defects.

No technologically intense industry, other than software, relies exclusively on product testing to remove defects. It has been known for more than 20 years that testing is an expensive and ineffective way to eliminate defects from any product, including software.

When leading manufacturers in other industries strive for high quality, they focus on the development and manufacturing processes. They use defined and measured procedures and established statistical methods. Many organizations routinely produce products that are, for all practical purposes, defect free. While they do not guarantee that their products are defect free, they do provide warranty and support services to minimize their customers' damage and inconvenience. They recognize that defects are not acceptable and, to the extent they can, they bear the full costs of fixing or replacing defective products. This is not true for software.

The State of the Art in Software Quality

Sadly, software suppliers do not generally take responsibility for the defect content of their products. They often even ship products that contain known defects, and they commonly charge customers for a significant part of the cost of fixing these defective products. The public is increasingly aware of and unhappy with these practices. Software is routinely blamed for common problems in almost any industry that serves the public, and the public expects software to perform badly.

When an industry gets a reputation for poor-quality products, it is in a risky position. Almost any serious problem could trigger a severe political reaction.

The typical argument for current software practices is that no one does any better. This, however, is wrong. A growing number of software organizations now follow sound management and quality practices to consistently produce high-quality products. They do this by improving the maturity of their processes and using sound engineering methods. These methods have been proven in other industries, and their benefits are now widely recognized for software.

The Public's Need

As long as software was used only in scientific or support applications and as long as lives and public well-being did not depend on software, software-development practices could be undisciplined and still cause little harm or disruption. With software becoming more critical, we face a key question: "Will the software industry address these needs voluntarily, or must the changes be

forced?"

What the public increasingly needs, and will soon demand, is the ability to easily distinguish between quality software products and defect-prone and unsupported work. They could readily understand the difference if products were clearly labeled. Public demand for quality products would then, in time, lead to enriched competitive offerings.

As long as members of the entire software industry insist that quality is not their responsibility, quality cannot be a competitive issue, and the public's interests cannot be served. It will only be a matter of time before some event demonstrates that the software industry's current software quality position is intolerable. When poor software quality poses major economic or life-threatening problems, we can expect an almost instant political reaction.

To gauge the likely consequences, compare the freewheeling practices in the software industry with those in such regulated fields as nuclear power, medical instrumentation, auto safety, or drugs. No software group would voluntarily work under the constraints common in these industries.

A Suggested Approach

I suggest the following. Make it clear that quality is the normal responsibility of the suppliers of software products and services. Require software suppliers to stand behind their products.

Permit any supplier to continue with the no-warranty and no-guarantee policies of today only under specific conditions. Such software products could be offered on an as-is basis with no quality obligations only if they were clearly labeled "AS IS, UNSUPPORTED, AND USE AT YOUR OWN RISK." With this label buyers would know that buying and using such products entails risks, and they could, therefore, make rational choices.

When software quality becomes an important customer concern, competitive market forces will, in time, ensure that the public's needs are addressed.

Suggested UCC Article 2B Changes*

While the industry is unlikely to change merely in response to Uniform Commercial Code (UCC) changes, it is important that current industry practices not be enshrined in the draft code or in the laws of the 50 states. We should make it clear that software suppliers are expected to produce quality products and to stand behind them. While exceptions must be permitted where undisciplined and low-quality work is acceptable, such practices should not be the industry norm.

This approach has the following implications. We should stop talking about software bugs as if they were mere annoyances. They are defects and should be so labeled. When a software product has one or more defects, it should be recognized as a defective product. Suppliers of defective software products should be expected to fix the defects and to minimize or remedy the damages the defects cause.

Conclusions

While today it may seem rational for the software industry to disclaim all responsibility for the quality of its products, this is tantamount to insisting that the market change before the industry will. This stance guarantees that when the market changes, as it must, the public must first be damaged. This will cause avoidable harm and will discredit present suppliers.

Continuing this strategy will mean that software quality will inevitably become a hot political issue. Not only will the politicians demand action, but they will not likely look to the industry that caused the problems to recommend solutions. In the past, such conditions have led to protective regulation to guard the public from irresponsible industry practices.

While it is not clear that such a reaction can now be prevented, an important first step would be for the industry to establish a responsible position regarding product quality. Even though the proposed changes would not likely cause immediate practice changes, they would establish a responsible public attitude. They would also provide a clear means for distinguishing high-quality offerings from all others.

While this may not seem advantageous to the U.S. software industry today, many developing countries have made software their highest industrial priority. They are actively pursuing the software business, and their principal target is the U.S. market. To maintain a sound competitive position, current U.S. suppliers need to differentiate their products in ways that are meaningful to their customers.

Quality is a proven way to do this.

* COMMENT: The UCC has been mostly adopted into law in most states. A new portion of the UCC - Article 2B - is being developed to cover the software and computerized information industry. That draft article, in its current form, is considered harmful to consumers of software. It must be stopped and kept from becoming a law in the State of Texas or the authors must be forced to address the issues raised by Humphrey in this article. Herb Krasner

NOTE: For references, for more information, or to get involved, please contact Watts Humphrey at watts@sei.cmu.edu

SQI to Offer Personal Software Process Training for Software Developers in '98

Here's the problem: the demand for new software professionals is growing by more than 500,000 per year worldwide. Only a fraction of that number (40,000-50,000), however, is being filled by qualified students graduating from U.S. colleges. The result may be a staff shortage that forces companies to delay important projects.

In the U.S. alone, there are currently 190,000 unfilled software positions, not counting small businesses, government, and nonprofit employers. This shortage is predicted to continue well into the 21st century.

What's the Solution?

One strategy that has emerged in recent years is to increase the productivity of software developers currently employed.

Headed up by Watts Humphrey of Carnegie Mellon's Software Engineering Institute and known as Personal Software Process (PSP) training, the method is a disciplined approach to software development that encourages developers to rationally measure their performance. This enables them to evaluate changes in the way they work and to steadily improve their practices. PSP has been shown to achieve a 5- to 10-fold increase in software productivity and quality.

PSP training shows software engineers how to:

- define their personal processes;
- estimate, plan, and track their work; and
- produce defect-free software projects.

Results from PSP trained staff have shown:

- improved estimating and planning;
- sharply fewer defects; and
- higher productivity.

SQI Seminars in PSP

In an effort to share this solution with others, SQI will offer PSP training through seminars starting in early 1998. Instructor for the PSP courses will be Jim Terrel, an SEI-trained instructor.

Watch www.utexas.edu/coe/sqi for details or call (512) 471-4922.

2nd Annual Telecom Conference Slated for Next February in Austin

Dr. Raymond Leopold, one of five founders of the IRIDIUM project, which expects to have 40 satellites in orbit by early next year, will be one of the distinguished speakers at the Second Annual Telecommunications Conference set for February 2-4, 1998, in Austin. Dr. Leopold is vice president and chief technical officer for the Motorola Satellite Group.

Other featured speakers at the conference include Dr. Pallab Chatterjee, chief information officer and senior vice president of the Semiconductor Group at Texas Instruments, and Dr. Mario Gonzalez, vice chancellor for Telecommunications and Information Technology for The University of Texas System.

Conference sessions will feature nationally recognized speakers on a spectrum of topics ranging from public policy for electronic commerce to cabling, network security, using the web for call center connections, distance learning networks, and much more.

The annual event is sponsored by UT's Department of Electrical and Computer Engineering, in cooperation with 27 leading telecommunications companies nationwide, and managed by SQI staff. See www.utexas.edu/coe/sqi for more details.

NASA's Shuttle Software: Lives Depend on Its Quality

by Paul Ehrler

Almost four years ago, the Austin Software Process Improvement Network (A-SPIN) featured a guest speaker from NASA's Space Shuttle Onboard Software Project in Houston. At the time, the project was run by IBM Federal Systems Company, one of the first organizations in the world to reach Capability Maturity Model (CMM) Level 5. Over the intervening years there have been two changes of ownership, but the project is still considered by many to be the standard for other organizations that strive for a high level of process maturity.

To learn more about how such a high-maturity group evolves over time, A-SPIN asked Ted Keller, project coordination manager for the Space Shuttle Orbiter Avionics Software Project, which is run by the Lockheed Martin, to address the group this summer. He presented "CMM Level 5: Three Years Later."

Extremely Dense Functionality

The onboard shuttle software serves as the control system for all flight phases of the shuttle plus ground-based activities such as preflight checkout and training. As such, it is among the most complex software systems in existence. The nearly two million lines of code (about a quarter of which fly on the shuttle) contain extremely dense functionality, including a proprietary real-time operating system. To give you some idea of just how dense the functionality is, consider that the requirements documentation runs to around 40,000 pages.

Literally hundreds of subsystems are continuously monitored over 24 data buses. Real-time constraints are stringent because the shuttle is inherently unstable in flight, anomalies must be detected and corrected before the human pilot even perceives that a problem exists. Both hardware and software fault-tolerant designs ensure safe operation in a hostile environment.

As impressive as this functionality is, the real significance of the onboard shuttle software is the extremely high level of quality that is needed. A single defect in an otherwise flawless system could result in disaster. Before every shuttle launch, Keller travels to Kennedy Space Center in Florida to sign a statement asserting that it is essentially error free and can be trusted with the lives of the crew.

Process Adherence

The key to any life-critical system is what Keller terms process adherence. Testing obviously plays a central role in producing a quality product, but since we know that exhaustive testing of any nontrivial software system is impossible, we must put into place "a defined, controlled, repeatable, predictable process that will result in a product of known quality," Keller said. That may sound like it was taken directly from the CMM, but in fact the onboard shuttle software project has been used by Carnegie Mellon's Software Engineering Institute (SEI) and ISO to help establish and refine standards like the Software CMM and ISO-9001.

Since few of us will ever work for a Level 5 organization, one might ask "so what?" As interesting as the onboard shuttle software might be, is it relevant to the more mundane real world? Marketplace realities prevent ordinary companies from acquiring the vast resources NASA has at its disposal. Believing that high levels of process maturity apply only to special cases like NASA, however, is missing the point.

Process Maturity: a Potential Resource

Quality is important in every effort, but you must consider the tradeoffs among quality, cost, and schedule. As Keller puts it, "You can only optimize around one of those three. You have to manage the other two." In other words, even though your particular enterprise does not require the extremely high quality of NASA, the same principles of process improvement can apply to reducing development cost or shrinking time to market. He cited Microsoft as an example where schedule might be the optimizing factor, with quality and cost managed. Seen this way, process maturity is a potential resource that can be used in many ways. In its absence, something is being wasted.

Another way of showing how principles of high quality can apply to ordinary organizations is by pointing out that you do not have to strive for the extremes that NASA requires. Quality is not an all-or-nothing concept, and any company can profit from process improvement, however modest the effort. The law of diminishing returns applies to quality improvement efforts, and the point of maximum payoff is different for each project. As Keller put it, "You can buy it by the yard."

Metrics is the Key

What characterizes a Level 5 organization? Perhaps the most salient aspect is having the ability to know and understand what you're

doing, why you're doing it, and what to do when problems are encountered.

Clearly, metrics is a key to that knowledge, and the Space Shuttle Orbiter Avionics Software Project, as expected, places a big emphasis on metrics. Configuration management is exercised at an amazing level of detail. Each individual line of code has a pedigree, a complete history of what changes were made over its lifetime. That may seem like a lot of record keeping, but in situations where there is no margin for error, detailed usable data are essential.

The Ability to Adapt: the Heart of Level 5

People issues are very important at Level 5. The concept of blame has been effectively banished: a discovered error is considered an opportunity for improvement. The project's environment encourages the discovery of problems. Every error or defect, no matter how slight, is formally investigated, and the goal is to desensitize the process, reducing the likelihood that the same or a similar error will recur. This ability to adapt lies at the heart of CMM Level 5.

Cultivating an environment like this is not easy. Keller spoke of balancing empowerment with the notion of desensitizing processes from human frailty. For a high-maturity organization, this is an effort that never ceases.

The primary goal of the project was not to reach CMM Level 5 or to become ISO certified. Instead, project members used both of those standards as guidelines to help attain the real goal: the highest quality possible. Focusing on a number or a certification can only distract an organization from the true objectives of process maturity.

Much has been written about the NASA project, including an entire chapter of the SEI book *The Capability Maturity Model: Guidelines for Improving the Software Process*. After listening to Ted Keller, high levels of process maturity seem a little more relevant to real-world organizations like yours and mine, and maybe a little more attainable.

How Software Development Is Like the Oil in Your Car

by Celeste Armstrong

We've all heard about the Software Engineering Institute's Capability Maturity Model for Software. And we know about its five levels of maturity and the 18 key process areas that characterize the software process of organizations.

But, have you ever thought of the five maturity levels in terms of, well, oil?

Level 1: No Dip Stick, No Owner's Manual

No dip stick and no owner's manual? That's okay, you'll do the right thing when the time comes. Level one maturity is like owning a car with no dip stick. You know the basics; you have a combustion engine, and it needs lubrication and coolant. Without a dip stick, though, you just drive your car while because it feels like the right thing to do.

You do remember, however, reading an owner's manual sometime from another car. It was for a different model and year, but you use it as a model of what kind of oil to add and approximately how often. You don't bother to write down what kind of oil you used or the date that you changed it last because you figure you'll remember something as important as the lubrication system in your car.

This is an ad hoc process.

Level 2: Writing It Down

After your Level 1 car's engine seizes and makes you carpool for a while, you move on to Level 2. You decide that you'd better have a dip stick in your next car and maybe you will start writing down when you last changed the oil and what type of oil you used.

When you get your next car, you look at the dip stick periodically, and you even get your significant other to look at the stick with you (your quality assurance), but there are still no gradations on your stick, and, while you are writing down the information, you are not quite sure if it is the right way to do it. You do have a record, though, so if you are successful in keeping your lubrication system going, you could repeat the process. This is a Level 2 car!

Level 3: Hey Look! Manual and Gauges!

You did pretty well with your Level 2 car. It sure lasted longer than the Level 1 car did, but you had to do some extra maintenance on it because you didn't have the owner's manual (it was a used car) and you put the wrong weight oil in it. You also didn't have gradations on the dip stick, so when you thought it was full,

it was really a quart low. Now you are having to spend a bit of money to maintain the car.

So, time flies and you get into your Level 3 car. Since you want the lubrication process to be more defined, this time you make sure that you have an owner's manual and that there are clear markings on the dip stick. You even get some training on how to maintain your car's lubrication system, and you are keeping good written records too.

Your car hums now when it runs down the road, and you even have a red light on the dash to tell you if your oil pressure has failed. You talk with your mechanic and take your car in to the shop to test the integrity of the engine periodically. You are doing pretty well with this car, and you think that things just couldn't get any better than this, but . . .

Level 4: How Long Since You Changed Your Oil?

Your neighbor has a Level 4 car. It has a pressure gauge on the dashboard, in addition to the right owner's manual, written records, and maintenance checks (peer reviews). Your neighbor is better off than you financially, so he can afford a few more bells and whistles on his car, and now he has a new model with electronic reminders about the lubrication system.

Wow! This guy has all you have in your Level 3 car, with a few extra twists. You're pretty happy with your Level 3 car, but your neighbor spends less time maintaining his car, and his car's operation is a little simpler. The gauge on the dashboard makes it easy to know when to add or change oil and tracks itself so the hood doesn't have to be opened so often. This Level 4 car is managing its maintenance for its owner, but . . .

Level 5: Your Oil is Low, Shall I Add Some?

Mr. Z founded a company and he is even better off than your neighbor! The sky is the limit when it comes to his car, and HIS car constantly monitors, measures, and provides feedback to its lubrication system.

Everything that is in your neighbor's expensive Level 4 car, your modest Level 3 car, and the basic Level 2 car is present in this car, but it is always running at its peak and it spends very little time in the shop for maintenance. When it determines from its various measures that the engine is not running at its best, it self-adjusts its systems to compensate and correct itself! It even carries around spare oil in an automated dispenser in its trunk so that Mr. Z doesn't even have to pull off the road for some of the maintenance needed.

This Level 5 car has everything the other cars do, but it optimizes the system as it goes, self-correcting itself as necessary. Of course, not everyone has the money to put into a Level 5 car, and even if they did, it would be difficult to own one if you didn't already have experience with the Level 1 through 4 cars: rather like having an expensive sports car, with no driver's license. It sure would look good, but it wouldn't be very practical.

So, What Kind of Car Do You Drive?

Everyone would like to see their software projects run like Mr. Z's Level 5 car, but it is rare to see a car like that. In fact, most cars YOU know look like that Level 1 car. You'd like your car to be Level 3, but your whole team has to drive the same kind of car, and some of you just can't afford it. Maybe if you all drive a Level 2 car for a while, you can trade up to a Level 3.

At any rate, no one can tell what level car you have by looking at the paint: no one has a bumper sticker that says, "we're number 5, we try harder!" You know what level your car is by how smoothly it operates and how you get to your destination on time, on budget, and with quality.

Gentlemen, Start Your Engines!

It is important to note that the level at which you are operating is *not* important. The only important result is *how* your work product is developed.

Can you make it around the racetrack of time, quality, and budget ahead of your competitors? More mature organizations produce output in a shorter time and with less staff than less mature organizations. That is because they know what to expect on a project and how to lubricate it for best results.

Remember the cooler the temperature, the lighter the oil.

Austin to Host IEEE International Multimedia Conference this Summer

The 1998 IEEE International Conference on Multimedia, set for June 28-July 1, 1998, in Austin, offers opportunities for local

companies to spotlight their products, services, and research through workshops, demonstrations, technical presentations, tutorials, and exhibits.

More than 400 are expected to attend.

Gary Cobb of Dell Computers, conference co-chair, notes that a key objective of the conference is to create a program that achieves a balance between theory and practice, academia and industry, systems and tools-oriented research, and content creation. Several Austinites are serving on the conference planning committee: Harrick Vin of UT's Department of Computer Science, Jim Babcock and Bryan Fugate of J&B Imaging, and Don Shafer and Christi Peterson of Crystal Semiconductor.

For details check out SQI's web page (www.utexas.edu/coe/sqi) or call (512) 471-4922. The event will be managed by SQI staff.

Successful President/CEO Peer Group Marks Good Start to ASC Initiative

by Karen Jonson

They are the leaders of several of Austin's most dynamic software companies, and yet many of them barely knew each other. Now more than 12 presidents and CEOs of Austin-area software companies with annual earnings over \$5 million not only know each other, but they're working together on issues important to their companies and to the local software community.

Earlier this year the presidents and CEOs of software companies such as Dazel, Pencom, VTEL, Evolutionary Technologies, and National Instruments joined the President/CEO Peer Group Initiative started by the Austin Software Council (ASC). Created in response to a 1996 member survey, ASC peer groups are smaller, more private and narrowly focused subgroups where software peers can discuss issues, share ideas, and trade solutions. The President/CEO Peer Group is the first of four peer groups ASC plans to form.

An Excellent Forum

ASC leaders describe the President/CEO Peer Group as one of the most successful, productive, and useful programs the council has initiated in the past year. Participants view it as an excellent forum for tackling common issues important to all members of the group.

"Providing a common forum, the President/CEO Peer Group allows software leaders to focus on shared business concerns such as staffing, technology changes, and pending legislation," says Bill Broussard, ASC program initiative chair for peer groups and a lawyer in private practice. "Coming together as a group also gives them the power to affect shared community goals."

"We started this peer group by asking 25 leaders in service industries such as law, banking, and accounting to recommend presidents and CEOs of companies earning over \$5 million annually," said Cerise Blair, ASC executive director. "More than 100 presidents and CEOs were identified. Of these, 21 were invited, 19 responded, and 14 attended the first meeting. The response was amazing, especially when you consider what these people are responsible for back at their offices," notes Blair.

Starting at a Gallop

In typical president/CEO fashion, group members wasted no time in targeting their key issues, establishing a strategy to address them, and implementing a results-oriented plan. The first issue they identified was Austin's tight high-tech labor market. The group decided that an important step in affecting change in this area was to learn about UT's technology curricula and programs.

By the second meeting, they were being introduced to UT by two college representatives. At the third peer group gathering, members spent five hours exploring technology programs, curricula, and future projects with several UT technology deans and directors.

"The leaders agreed that they would like to provide input on UT's curriculum in order to help ensure that graduates entering the job market have the right skills and training," says Steve Vandegrift, ASC chairman and president and CEO of Activerse, Inc.

In addition to making in-roads with UT, members of the President/CEO Peer Group have also begun to investigate ways in which the organization can work with the Greater Austin Chamber of Commerce for the mutual benefit of both groups.

"We share many of the same problems in attracting new employees, finding facilities, working with community agencies more accustomed to dealing with large organizations (such as Motorola, IBM, Samsung, etc.), and competing with companies headquartered in Boston or Silicon Valley," says Bill Bock, CEO of Dazel Corporation. "This peer group allows us to tackle important issues in a way we could not as

individual organizations. Our efforts to improve the relationship between the local software industry and UT and the Chamber of Commerce are simply two early examples. There will be others."

More Peer Groups in the Works

The President/CEO Peer Group is just the first of four peer groups ASC envisions. The Entrepreneurial Peer Group is the second group to be formed. It is for CEOs and CO's of software start-up companies. Dale Cox of Price Waterhouse is in charge of getting the group started. Sixty entrepreneurs have already been invited.

The third peer group will be for marketing and sales executives at large software companies. It is scheduled to get underway before the new year. The fourth peer group will be for emerging leaders of software firms with less than \$5 million in annual revenue. ASC is still looking for people to organize these two peer groups.

Hallmark of Success

Regarding the success of ASC's first peer group initiative, Broussard says, "the level of participation is the most important indicator of the group's value. Despite their busy schedules, these presidents and CEOs have found time for these meetings. In fact, no other event has drawn as many high-tech leaders together at one time and place. It's phenomenal," notes Broussard.

Additional Information

To learn more about ASC's peer groups or to help plan future initiatives, contact Cerise Blair at cerise@ati.utexas.edu or 305-0032.

About the author. *Freelance writer Karen Jonson has been writing for and about the high-tech industry since 1982. She specializes in high-tech marketing and public relations.*