



PACIFIC NW  
28TH ANNUAL  
SOFTWARE  
QUALITY  
CONFERENCE

OCTOBER 18TH – 19TH, 2010



ACHIEVING  
QUALITY  
IN A COMPLEX  
ENVIRONMENT

*Conference Paper Excerpt  
from the*  
CONFERENCE  
PROCEEDINGS

---

Permission to copy, without fee, all or part of this material, except copyrighted material as noted, is granted provided that the copies are not made or distributed for commercial use.

# Inspiring, Enabling and Driving Quality Improvement

Jim Sartain  
jsartain@adobe.com

## Abstract

This paper discusses the approach used at Adobe Systems for driving continuous quality and engineering process improvement through the adoption of engineering best practices including Team Software Process (TSP), Scrum, Peer Reviews and Unit Testing. It covers what has worked well and some challenges encountered. Key success factors and an overall methodology for driving improvement are outlined including:

- A multi-year improvement approach that can work in large organizations that may include both eager adopters and resisters of process improvement.
- Customer Satisfaction and Defect Removal metrics, including what is world class, what is typical, and the best practices that can drive major quality and engineering process improvement.

## Biography

*Jim Sartain is a Senior Director responsible for Software Quality at Adobe Systems (makers of Photoshop, Acrobat, and other industry leading technologies). He leads a team responsible for inspiring, driving and enabling continuous quality improvement across Adobe world-wide. Prior to Adobe Systems, Jim held quality/engineering process improvement and software development leadership positions at Intuit and Hewlett-Packard. While at Intuit, Jim drove significant improvement in quality and software engineering practices across the company. His last job at HP was in the role of CTO/CIO for an Airline Reservation Systems business that serviced low-cost airlines including JetBlue and RyanAir. Jim received a bachelor's degree in computer science and psychology from the University of Oregon, and a M.S. degree in management of technology from Walden University.*

Copyright Jim Sartain, August 2010

# 1 Company Background

Adobe Systems is a leading provider of solutions that enable customers to create rich and engaging internet experiences. Headquartered in San Jose, California, Adobe had close to \$3.0 billion in revenue in fiscal 2009 with greater than half of this revenue generated outside the United States. In 2010, Adobe Systems was ranked #1 on Fortune Magazine's list of the world's most admired software companies.

Adobe has more than 8600 employees with software development in many countries including the United States, India, China, Canada, Romania, Germany and Japan. Popular products include Photoshop®, Premiere Pro®, Acrobat®, Dreamweaver®, InDesign®, Flash Authoring® and LiveCycle®. Adobe has some ubiquitous products including Adobe® Flash® Player and Photoshop®. Flash® is on 98% of connected PC's and has 8 million installs per day. More than 90% of creative professionals have Adobe Photoshop® software on their computers.

# 2 The Business Opportunity for Quality Improvement

Some software development teams spend a significant portion of their development effort on defect driven rework. Software development activities beyond when software is declared "functionally complete" – a point at which teams are primarily focused on finding and fixings bugs is typically rework. I've seen some teams at HP, Intuit or Adobe Systems consume up to one-third of their overall development effort on testing and fixing bugs with another 15%-20% of effort required post-release to deliver incremental "dot" releases that repair shipped defects.

Organizations employing quality-first software development approaches can reduce these rework costs to less than 10% of overall development effort. These teams deliver software on a regular basis (e.g. monthly) that is at or near release quality and these releases do not typically require subsequent dot releases to address quality issues.

Another significant driver of software quality costs is customer service expenses. Often, a majority of customer service costs are driven by usability, reliability or performance issues. The customer experience can be improved and support costs significantly reduced by identifying and eliminating the root causes for the most common customer issues.

The direct costs of poor quality (e.g. engineering rework, customer care costs) are usually the tip of the iceberg for quality improvement opportunities. Westinghouse Electric Corporation found that their indirect costs of quality (e.g. reduced sales, less time for innovative work) were three to four times the directly measured costs of poor quality (Campanella 1999).



Figure 1: Quality Improvement Opportunities



## 4.1 Customer Benefits

The vision should emphasize directly engaging with customers utilizing techniques such as NPS to maximize the number of product promoters. Adobe's goal is to have 70% of its customer be promoters. Adobe has found that its most satisfied customers:

- Are more likely to recommend Adobe products to others
- Invest a greater percentage of their budgets with Adobe
- Are less likely to consider alternative solutions

## 4.2 Employee Benefits

Employee benefits should include improved work-life balance. Also, employees will benefit from having more time available to do valuable work. Providing a less stressful work environment that provides more time to do truly innovative work is a great way to increase employee engagement and ultimately retention. Some Adobe teams have freed up 20% more time from reductions in defect-driven rework.

## 4.3 Shareholder Benefits

Improving product quality can increase the number of product promoters and decrease the number of detractors. Increasing promoters and decreasing detractors will lead to increased sales, customer retention and the ability to charge premium prices.

Productivity improvements from early and efficient removal of defects can free up capacity for work that has a higher return-on-investment. Organizations adopting a quality first paradigm can avoid a vicious cycle where they must spend an increasing amount of their efforts on customer support, software maintenance and bug fixing. Because they are spending an increasing proportion of time on those activities they may not be able to direct adequate resources to innovation. This can ultimately lead to a lack of competitiveness, business decline or even bankruptcy (Humphrey 2001).

Quality improvements will also benefit business efficiency. Customer care budgets can be decreased by reducing the number of customer service calls driven by software defects, usability issues or overly complicated business processes. Other business functions (e.g. legal, PR, engineering) are frequently impacted with the emergence of and aftermath from quality issues.

# 5 Required Leadership Support

Most software engineers have a strong desire to deliver software of high quality and will if they are provided the right environment. This environment must include strong support for quality from all levels of management. To ensure strong leadership support for quality, managers should have quality and engineering process improvement as part of their performance goals.

An important leadership responsibility is to ensure delivering quality software that meets business goals is the highest priority for the organization. Business goals will typically include the delivery of software on a specific date. This is not a contradiction -- focusing on delivering quality software is the best way to ensure delivery dates are met. A frequent root cause for missing schedule commitments is finding an unexpectedly high number of defects late in the development cycle.

## 5.1 Quality Improvement Objectives, Goals and Metrics

A Quality Plan including goals and a strategy for achieving them should be established at the start of a project. Teams should have compelling quality improvement goals that include specific measurable goals and realistic improvement targets. Improvement targets should be set by the teams that own accomplishing them to ensure ownership and commitment for achieving them. If targets are set tops-down by managers then there is a risk of the teams not buying into them because they don't "own" them. In some cases, teams will believe their tops-down targets are impossible to achieve. Any lack of confidence or commitment may not be discovered until it is too late to prevent a schedule slippage and/or release quality issues. The quality goals must be clearly

aligned with what is important to the organization. For example, tying a goal to improve productivity to providing the opportunity to do more important and interesting work can be more relevant to employees and more likely to gain their support.

To help set aggressive but achievable goals it is useful to have benchmarks. The best benchmarks are from other teams within the organization. The next best source of benchmarks would be from other organizations in the same enterprise. If this isn't possible then industry data can be used (Jones 2010). Benchmarks closer to the team are not only more likely to be relevant; they are also more likely to be accepted by the team as valid. The collection and sharing of metrics across an organization will help provide relevant benchmarks. These benchmarks can help support a cycle of teams learning about best in class performance in the org and then raising the bar when they set their next set of improvement goals.

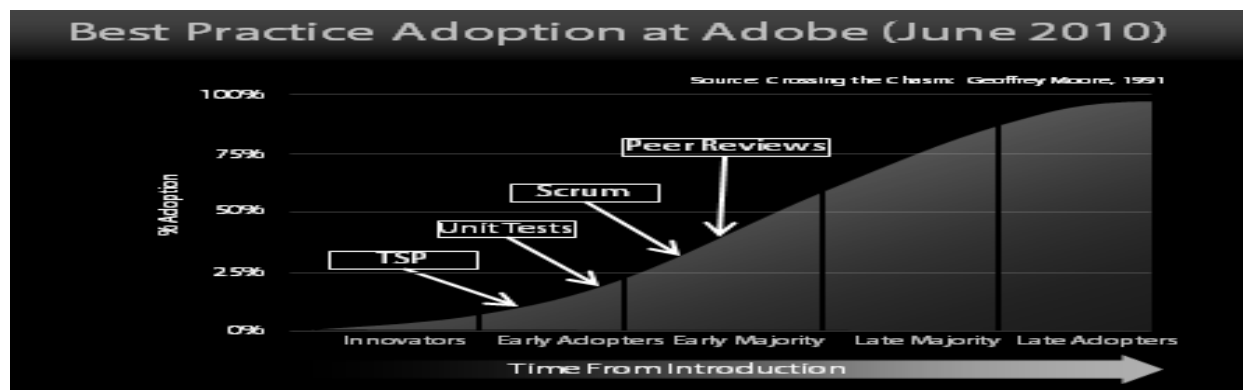
## 5.2 Quality Review Meetings

The senior leadership team must regularly review quality as a team. During these review meetings each team leader summarizes improvement progress and status. This provides an opportunity to recognize teams making progress. By publicly recognizing progress leaders reinforce the importance of continuous improvement. In cases where progress is not being made there should be discussion as to why and commitment and the necessary support to get improvement on track. Senior leader's regularly spending time discussing quality goals and improvement progress will highlight their importance. Also, these meetings serve as an educational opportunity for business leaders to better understand software quality economics. This deeper understanding enables the business leaders to ask the right questions and to be more specific (and genuine) in their praise about improvement progress.

## 6 Engineering Best Practice Rollout Strategy

Most engineering best practices have been around for years and are known to many engineers as best practices. Why aren't they common practice? Teams must take risk to do their work a new way. They must invest extra effort to learn new tools and techniques and usually need training and consulting on how to do them well. Best practices usually require support from the entire team. Unless there is strong leadership support for the need to change, one or two individuals on a team can veto or otherwise obstruct improvement.

The level of motivation for adopting change normally varies from team to team, from highly receptive to resistant. It is a bad idea to force teams to adopt a best practice. Teams forced to adopt a new practice are rarely effective at doing so. Adobe uses a viral approach where teams are encouraged to be pioneers. When early adopters demonstrate results their initiative, risk-taking and results are publicly recognized. These teams then become promoters. The most credible promoter for a practice is a well-regarded engineer versus a senior manager or quality/process improvement leader. Engineering best practice adoption at Adobe has followed a classic technology adoption curve as described by (Moore 1991). Figure 3 below illustrates the current level of adoption.



Scrum and the Team Software Process (TSP) are being used by teams to provide effective project management frameworks. Both methodologies have helped teams to become more effective at self-management as well as focusing on delivering quality code using iterative development. They both make use of early defect removal practices such as Peer Reviews and Unit Testing and tools for estimation and continuous learning from project team retrospectives. Based on the current rate of adoption, 80% of teams at Adobe will adopt either Scrum and/or TSP by the end of 2012.

Scrum is an increasingly popular methodology for managing projects using an agile development approach. A majority of teams adopting Scrum at Adobe demonstrated significant improvements in software quality, schedule predictability and employee satisfaction with their work process. Scrum is becoming the dominate and fastest growing approach to software project management at Adobe.

TSP is a methodology that provides training, support and tools to enable a team to be self-managed and to deliver on business objectives with exceptionally high quality and engineering productivity. TSP is described in Watt Humphrey's book "Winning with Software" (Humphrey 2001). TSP teams collect and use excellent software quality metrics to guide their projects. Further sections of this paper summarize results from Adobe TSP teams as an example of what is possible when quality plans and metrics are used to drive a quality-first strategy.

## 7 Metrics are key to Driving Effective Best Practice Adoption

A critical tool for driving engineering best practice adoption is the use of metrics to ensure effective use and to demonstrate their ROI. Figure 4 below shows a key effectiveness metric for defect removal best practices – the number of minutes of effort to discovery a defect by method. Defects discovered through personal reviews or peer reviews required an average of one hour of person-time to find and resolve, whereas the average defect found in system test required about eight hours.

It is best to use metrics that measure outcomes and not just activity. For example, having teams count the number of peer reviews they do is not sufficient. Even if lots of peer reviews are being done if they are not finding defects then they are not accomplishing their purpose. What matters most are metrics like the % of defects removed using early defect removal methods. Ideally, defects are removed in the same phase they are injected when they are frequently an order of magnitude less costly than the next phase. Quality cannot be "tested into" a product.

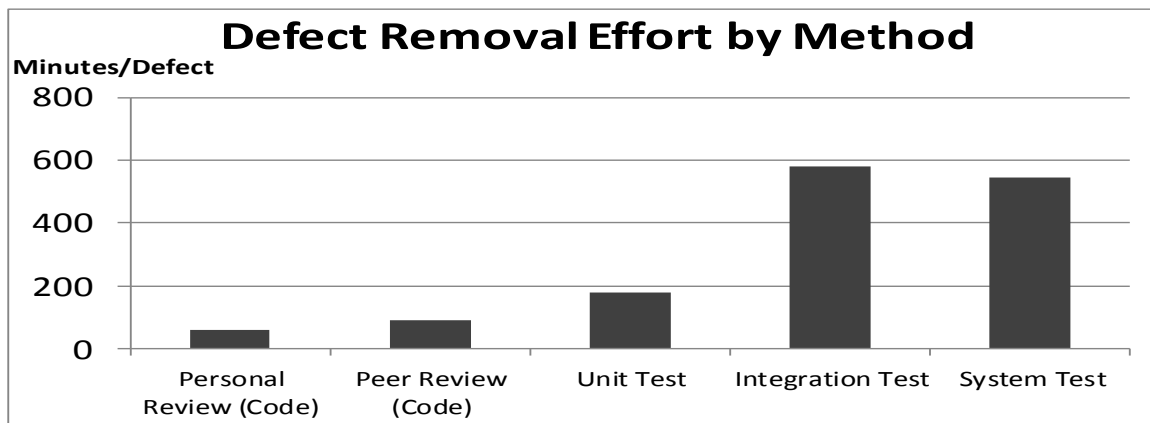


Figure 4: Defect Removal Effort by Method – Adobe TSP Projects

The data from Figure 4 was collected from Adobe TSP projects. Many Scrum teams are starting to collect this same data as a way to ensure these defect removal practices are being done effectively.

## Reducing Defect Driven Rework

A benefit of removing most defects prior to System Test is a significant reduction in engineering rework. Figure 5 shows the percentage of defects removed before System Test for a sample of seven Adobe TSP projects. TSP projects establish and manage goals and plans for removal of defects at each phase of the software engineering lifecycle. They also establish quality plans where they estimate the number of defects that will be injected and removed in each software development activity (e.g. design, coding) for each major component. These plans are used to assess whether a team is being efficient and effective in using early defect removal techniques.

Project	% of Defects Found Early	%Effort in Post-Dev Testing
A	94%	11%
B	83%	15%
C	75 %	16%
D	78%	32%
E	88%	18%
F	83%	9%
G	75%	13%
<b>Average</b>	<b>82%</b>	<b>16%</b>

Figure 5: Team Software Process Early Defect Removal Results

This reduction in post-release rework translated directly into these teams having more time available for value-added work (see Figure 6). In this chart the total time to find and remove defects through all methods is compared (Total Cost of Quality) as well as the average pre-system test removal rate (% of Defects Found Early).

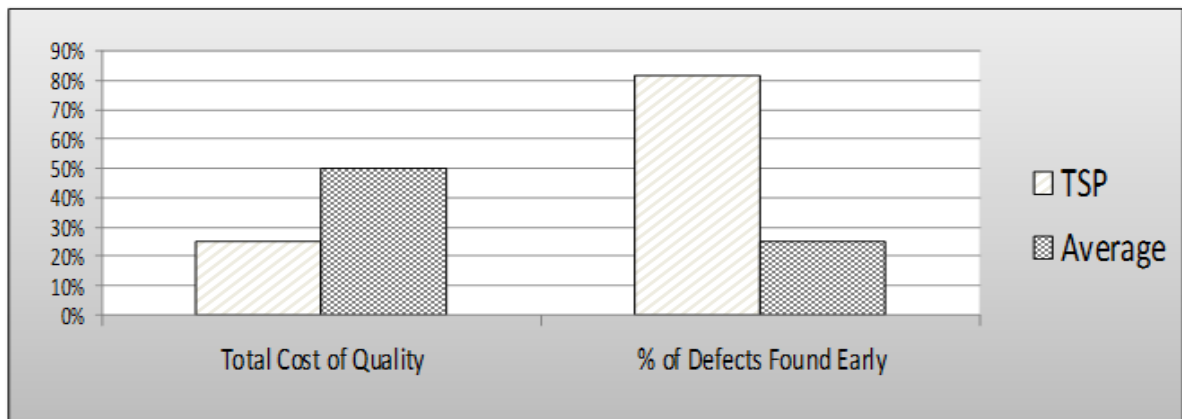


Figure 6: Team Software Process Early Defect Removal Results

These teams found a greater percentage of software defects prior to system testing. Defects found before system test are significantly less expensive to resolve. Also, the percentage of defects found before system test is directly correlated with post-release quality since System Testing will not find all defects. These teams were less frequently in fire-fighting mode and able to maintain more predictable work schedules and reasonable work-life balance. As a result, most of these teams had a larger portion of their development cycle available and allocated to design and implementation, versus being spent in a code-and-fix during system testing.

## 8 Summary and Conclusions

To deliver quality work and improve how it is done, the senior leadership of the organization must expect and encourage it. Senior leadership support includes ensuring managers have quality improvement as part of their performance goals as well as regularly review progress with the teams. These quality reviews are also an opportunity to ensure quality is a priority and that the necessary time, dollars and headcount for engineering quality into the software are available and used.

A key enabler of quality is the adoption of engineering best practices including Peer Review, Scrum, Team Software Process and Unit Testing. Self-directed teams adopting TSP and/or Scrum at Adobe are able to ensure that quality, schedule, scope and cost were not strict trade-offs. Through a combination of better planning and estimation, improved project execution and effective use of quality best practices such as unit testing and peer reviews, teams are able to deliver high quality software, on schedule and budget, with good work-life balance for the team. Learnings from retrospectives and improved metrics helped drive continuous and significant improvement in product and process.

## Acknowledgements

I would like to thank Watts Humphrey and Noopur Davis for the transformative influence they have had on many of the teams that they worked with at Adobe and Intuit. Also, I've learned so much about software engineering best practices and how to make this stick from Watts and Noopur. Thanks to Barry Hills, Johnny Loiacono, Paul Gubbay, Dave Burkett, Bill Hensler, Winston Hendrickson, Karen Catlin and Kevin Lynch for their strong support of the Adobe Quality Initiative.

## References

1. Campanella 1999, *Principles of Quality Costs: Principles, Implementation and Use*, ASQ Quality Press.
2. Davis, N. 2003, *Team Software Process (TSP) in Practice*, SEI Technical Report CMU/SEI-2003-TR-014 (September 2003), SEI.
3. Humphrey, W. 2001, *Winning with Software*, Addison Wesley Professional.
4. Jones, C. 2010, *Software Engineering Best Practices*, McGraw Hill
5. Kotter, J. 1996, *Leading Change*, Harvard Business School Press.
6. Moore, G. 1991, *Crossing the Chasm*, Harper Business.
7. Reichfield F. 2006, *The Ultimate Question: Driving Good Profits and True Growth*, Harvard Business School Press. and <http://www.netpromoter.com>