

# Delivering on the Promise of Process Improvement

Bill Thomas

In 1995, the Software Engineering Institute introduced the Personal Software Process<sup>SM</sup> (PSP<sup>SM</sup>), a technology that asks software engineers to do their jobs an entirely different way. Through formal training, engineers learn to set personal goals for improvement, measure and analyze their work, and adjust their processes to meet their goals. They develop the ability to predict their performance and manage the quality of the work they produce. Organizations using PSP have reported significantly improved size and time-estimating accuracy, as well as reductions of 60–75% or more in the number of defects injected during development or found during unit test.

While the payoff is great in terms of improved quality and efficiency, so too is the required commitment. Often, PSP techniques only become ingrained after engineers use them consistently for several months. Also, PSP usually works best when entire teams are trained together to use the techniques. Recognizing these challenges, the SEI introduced an experimental version of the Team Software Process<sup>SM</sup> (TSP<sup>SM</sup>) in summer 1996 to address some of the issues that could prevent PSP from achieving complete success. Since then, several dozen TSP teams have been launched.

This article surveys some successful installations of PSP and TSP, and presents some lessons that organizations have learned along the way. As some of those lessons actually contradict each other, perhaps the first lesson is that PSP and TSP are evolving technologies that will work differently under different circumstances. (For an overview of PSP and TSP, please see the Background article in this issue of SEI Interactive.)

## PSP takes hold at Boeing

The Boeing Company's John Vu has enjoyed remarkable success implementing the PSP, but not without learning some hard lessons along the way. Vu, a Boeing technical fellow and chief software engineer, is responsible for software engineering process improvements company-wide. He says his early efforts to provide PSP training to individuals from around the company yielded few positive results.

“My number one lesson learned is that random training does not work,” Vu says. “If you have an open class, and you have different people from different groups, they may not be

able to apply PSP/TSP successfully.” Vu has surveyed participants from the “random training” classes to determine whether they were still using PSP. “Only 10 percent said they were. You must focus on a specific project or program, build an infrastructure, and train everybody. Then it works very well.”

In successful PSP training at Boeing, Vu has concentrated on training groups of about 25 engineers over six months. A full-time “coach-instructor-mentor” works as a member of each group’s staff, and helps with training and collecting personal data and statistics, which are anonymously recorded and converted to trend lines. “Without a coach, you can’t maintain the focus and direction of the program the way you want it,” Vu says.

The groups have delivered impressive results. Defects have dropped by almost 75% over past software releases. Delivery time is up only one-quarter of 1%, but the groups are writing programs that are 2.4 times larger than previous programs, and three to four times more complex. “We do it faster and we reduce the defects by close to 75%,” Vu says. “To me, the data speaks for itself.”

Vu has experimented with different approaches for providing training, and has settled on a half day per week for six months. “Most engineers say one-half day per week is very reasonable. They complain that they don’t have enough time for the homework, so we are trying to give them another half day for homework.”

The two current programs are pilots, but Vu says he will soon have enough data to justify a company-wide PSP program.

Vu, who is an SEI resident affiliate and has worked closely with the SEI technical staff since 1988, says that despite Boeing’s success, he has some concerns about PSP. “I believe you cannot use PSP/TSP before an organization has reached Level 3,” he says, referring to the SEI’s five-level Capability Maturity Model® for Software. (For more on the CMM® framework, see <http://www.sei.cmu.edu/cmm/cmms/transition.html>.) “At Level 3 or 4, the organization is more mature and has much better management support—and management support is critical.” In mature organizations, management understands that when a group of engineers, using PSP, provides a well-developed schedule for a project, the timetable is accurate, even though it may be longer than management would prefer. “If you don’t have a mature organization, managers will say, ‘I’ll give you one week, but not two.’ In more mature organizations, managers appreciate the data that engineers show them from PSP about the need for more time.”

Vu says he also believes that organizations that use PSP/TSP can accelerate to Levels 4 and 5 more quickly, and he is currently collecting data to support that belief.

## Motorola applies PSP principles

Motorola first introduced the PSP approach in 1995 when a group of managers teamed up to explore ways to improve software engineering practices and achieve CMM Level 3, recalls John Pang, a staff engineer who was part of the original pilot project and has been working with several PSP projects since. Motorola hired an instructor from Embry-Riddle Aeronautical University who taught two courses, one for managers and one for engineers, using the textbook *A Discipline for Software Engineering* by the SEI's Watts S. Humphrey.

Since that time, about 100 people at Motorola have been trained to use PSP, all in groups of coworkers. As at Boeing, Motorola experimented with the training schedule. Motorola has adapted the original schedule of 3 training days per week for 15 weeks, to 2 days per week for 10 weeks. The company does not provide trainees with time for homework, which they must do on their own time. The company acknowledges the engineers' extra effort by throwing a party for the workers and their families at the end of the course.

Pang says that his organization achieved its goal of Level 3 assessment because of three factors:

1. strong support from senior management, including resources
2. the vision of the manager of the software engineering process group (SEPG), who set Level 3 certification as the overall goal
3. widespread training in the Personal Software Process, which helped the organization understand what it needed to do to reach Level 3

The SEPG leader "knew what he was doing. He knew how to talk to senior management, middle management, and the engineers, and how to steer them to his vision of CMM Level 3. But we could not have followed his vision for CMM Level 3 without PSP, because the engineers could have seen the effort as a burden, as extra work. PSP provided the education. It opened our eyes to the fact that process improvement is possible, and why things like requirements, planning, and inspection are important." Later, when the process group needed to institutionalize inspections, "We said, 'OK, that makes sense. Let's do it,'" Pang says.

The engineers at Motorola found, however, that they could not strictly adhere to the Watts Humphrey textbook. Rather, they needed to tailor the course to their own needs and organization, Pang says. "We took the principles, such as 'inspection is valuable' and 'fix defects as early as possible.' So we asked ourselves, 'How can we institutionalize inspections? How can we manage ourselves? How can we negotiate estimates and schedules?'" Groups also developed their own training exercises. Rather than using an

example from the textbook, they built exercises around how to program functionality into a new line of paggers—an actual workplace project.

Pang declined to share specific data on improvements but, he says, “The engineers are much happier. They don’t have to go into the factory because a bug shut down the line. We like our jobs better than we did five years ago.”

### **Software engineers at AIS take to PSP**

Advanced Information Services, based in Peoria, IL, became interested in software process improvement in 1992, when the owner of the consulting company read some of Watts Humphrey’s writings on managing the software process. The owner asked Humphrey if the concepts could be applied to a small organization of only 35 employees, who worked on projects in teams of five or fewer engineers. At the time, Humphrey was developing the techniques that would become the Personal Software Process, and he met with AIS staff members to discuss his ideas. When the SEI taught its first instructor’s training course for PSP in 1994, AIS sent two software engineers to become instructors.

Today, AIS employs some 150 software experts at its offices in Peoria and Evanston, IL, and at a wholly owned subsidiary in Chennai, India. The company provides software consulting services in a wide range of application areas such as embedded systems, information systems, and e-commerce, covering technologies including the Internet, client-server networks, and mainframes, and supporting programming languages from C to COBOL to Oracle.

AIS made an early commitment to PSP for the members of the firm’s development group. “We knew that PSP was the way we were going to go for projects that we had control over,” recalls Robert Pauwels, commercial training director for AIS. The firm ran a pilot project, then introduced PSP more broadly. Today, the PSP is a required training course for all AIS software professionals.

Pauwels says that he believes PSP was easier to implement at a consulting firm like AIS, as opposed to a manufacturing company, because AIS consultants are already used to some PSP principles, such as tracking time. “Being a consulting firm, we’re used to time tracking for billing purposes. But detailed time tracking is extremely foreign to many organizations.” AIS also had a defined process for software development, and practiced peer reviews to promote early defect detection.

For AIS today, the challenge is to quickly train new hires to use PSP. “We do a fair amount of recruiting from the universities in this area,” Pauwels says. “We put recruits through the course as soon as we can. The ideas are still quite foreign to them, compared with what they’ve just invested four years of their lives in at the university. Most are surprised at what we’re asking them to do. But it is entrenched in our culture and institutionalized, and that helps break down the barriers of resistance that they often have.”

As at other organizations, PSP has shown powerful results at AIS. “One of our goals from an organizational standpoint is to produce software modules with zero defects, and we are leveraging the PSP skills to do so.” For example, on their most recent projects, 100 out of 148 modules had zero defects from integration test forward. In another example, AIS completed an 80,000-line Visual Basic project. The delivered defect ratio was 0.05 defects per 1,000 lines of code—a small fraction compared to the number of defects that would be found on a similar project for which PSP was not used.

Pauwels adds that PSP, and also CMM training, have greatly increased the speed at which AIS completes projects. “It used to be that our projects would take months to get through system test. We’ve been able to squeeze that down to days now by putting a focus on defect detection throughout the development process. In the early days, we had to wait until test to get our first indication of product quality. By coupling PSP and CMM together, early defect detection has become ingrained in our culture.”

### **Early Observations of TSP**

Since the SEI introduced the Team Software Process in summer 1996, several dozen TSP projects have been launched with commercial and government organizations. From those experiences, the SEI’s James Over has collected some “war stories” about TSP’s early implementation.

In one example, a team of nine software engineers and five hardware engineers were assigned the delivery of a critically needed hardware–software product in nine months or less. The team defined its development process, estimated the size of the job, developed an overall plan, produced detailed next-phase plans, balanced those plans, and assessed project risks and problems. This effort called for an 18-month delivery schedule. Under strong management pressure, the team explained its plan and justified the effort required. Management eventually accepted the plan and the team finished six weeks ahead of schedule.

On another team, defect ratios have dropped dramatically. Before the organization instituted the TSP approach, a project with 9,500 lines of code had an average of five defects per 1,000 lines. After TSP training, the team completed a project with 89,900 lines of code, which had .02 defects per 1,000 lines. That is an improvement rate of 250 times.

Where successfully implemented, the TSP approach “provides teams with detailed plans, the data to justify their commitments, and the conviction to defend their plans,” Over says. “In every case, management has accepted the team’s committed dates. In all cases where the team has followed the process, they have met their dates.” In addition, Over says, “When engineers use the TSP, they produce very high-quality products. This is

because they measure and track quality, apply quality methods, and have quantified quality goals.”

Still, not every TSP implementation has been a success. Since 1996, two TSP projects have failed, both because of major management shakeups that resulted in pressure to cut costs and widespread disruptions of engineering staff. Also, three projects have had only “so-so” results. Of those, none included management training and management did not monitor the teams’ performance. The engineers involved eventually stopped gathering and using TSP data and the projects did not follow the TSP process. Nonetheless, exposure to TSP made a difference: the projects all delivered working products, finished close to on time, and appeared to produce improved quality.

Over also considers some projects to be “shaky.” “The shaky projects have many part-time engineers, untrained management, or no coaching support,” he says.

### **TSP Success at Hill Air Force Base**

One of the most successful TSP implementations to date is at the Hill Air Force Base (AFB) software group, which develops avionics and support software for the U.S. Air Force. The group sent several engineers to the SEI’s PSP instructor authorization program. In a recently completed pilot project of 25,820 lines of code using the PSP and TSP approaches, the project team delivered the product a month ahead of the originally committed date, at almost exactly the planned cost. During system and operational tests, the team found only one high-priority defect and reduced testing time from 22% to 2.7% of the project schedule. This product is currently in acceptance testing, and the customer has found no defects to date.

“TSP gives you incredible insight into project performance,” says one member of the Hill AFB software group. “You can really see what needs to be done...I saw this group go from setting ‘square-filler’ goals to setting real, achievable goals, and now I see them achieving those goals.”

“TSP creates more group involvement,” says another Hill AFB engineer. “Everyone feels like they’re more part of a group instead of a cog in the wheel. It forces team coordination to talk about and solve problems.”

### **About the author**

Bill Thomas is the editor in chief of *SEI Interactive*. He is a senior writer/editor and a member of the SEI’s technical staff on the Technical Communication team, where his duties include primary responsibility for the documentation of the SEI’s Networked

Systems Survivability Program. His previous career includes seven years as director of publications for Carnegie Mellon University's graduate business school. He has also worked as an account manager for a public relations agency, where he wrote product literature and technical documentation for Eastman Kodak's Business Imaging Systems Division. Earlier in his career he spent six years as a business writer for newspapers and magazines. He holds a bachelor of science degree in journalism from Ohio University and a master of design degree from Carnegie Mellon University.

The Software Engineering Institute (SEI) is a federally funded research and development center sponsored by the U.S. Department of Defense and operated by Carnegie Mellon University.

- <sup>SM</sup> IDEAL, Interim Profile, Personal Software Process, PSP, SCE, Team Software Process, and TSP are service marks of Carnegie Mellon University.
- ® Capability Maturity Model, Capability Maturity Modeling, CERT Coordination Center, CERT, and CMM are registered in the U.S. Patent and Trademark Office.