

Synergy of review techniques from PSP¹ to Formal inspections

Daniel M. Roy

(STPP, Inc. - Software Technology, Process and People, Inc.)

(Visiting scientist, SEI)

Abstract: Observations, data, and an experimental framework based on the principles of the experience factory are proposed to show the synergy of the PSP personal review techniques, the TSP approach and Fagan's formal inspections. The model can be used to predict defect removal patterns at the personal, team, and organization levels.

1. Introduction

Since their introduction in 1976, formal inspections have had a highly positive impact on the maturity of the software process and on the quality of the software products of numerous companies world-wide [Gilb-93]. A large body of evidence has established the fact that formal inspections are one of the most cost effective and easiest measures that can be put in place to make an immediate positive impact on any organization involved in software development.

The Capability Maturity Model (CMM)² has had a profound impact on the organizational practices within the software industry [Herbsleb-94]. It is fitting that Peer Reviews feature preeminently in CMM V1.1 (been a level 3 Key Process Area). Incidentally, it is also totally incomprehensible that PR has been dropped as a KPA from CMMI-SE/SW, released for public use by SEI last August.

Others, even more general SPI paradigms such as the experience factory have been demonstrating the value of experiment (an model) based software improvement for over 20 years now [Basili-89].

Building on the success of the CMM, the personal software process was developed by Watts Humphrey [Humphrey-95] by downscaling CMM major practices to the level of the individual engineer. Using fairly simple and well proven engineering principles, the PSP trained engineer plans his/her work, enacts a well defined process, building the product while gathering data, and performs a post mortem analysis that seeds the next improvement cycle. In so doing, the PSP is much more than a downscaling of CMM V1.1. It can be seen, and it is taught by the author, as the downscaling of Basili's Experience Factory (fig. 1), the spirit of level 5.

¹ Personal Software Process, PSP, Team Software Process and TSP are registered trademarks of CMU

² CMM and Capability Maturity Model are registered in the U.S. patent office

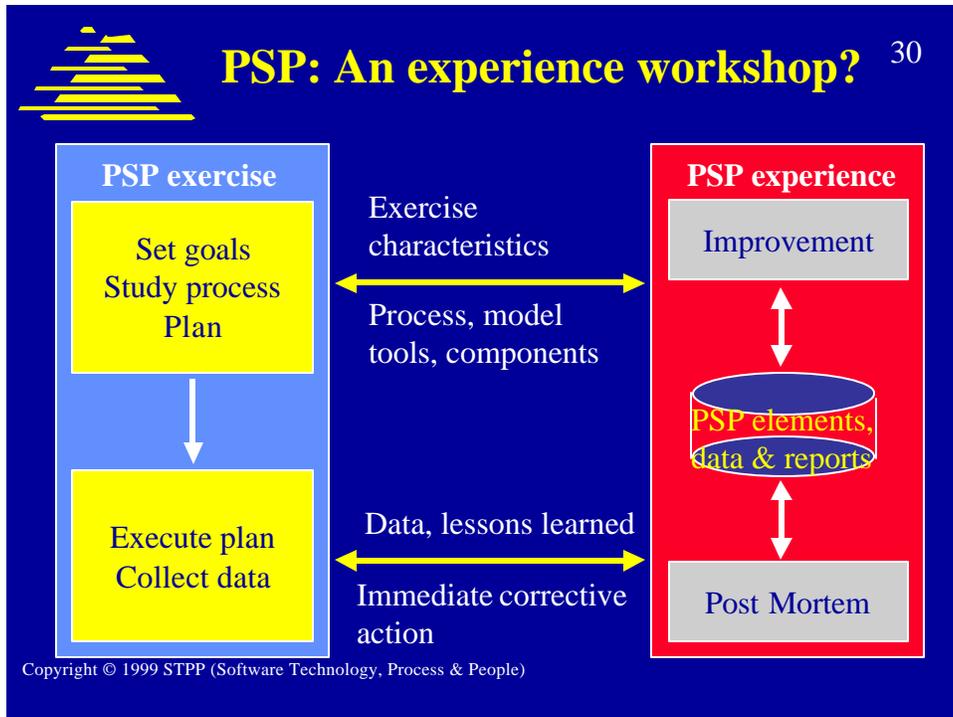


Figure 1: Downscaling the factory

It is fitting that personal design and code reviews (downscaled from Fagan’s inspections) feature preeminently in PSP. However, the current PSP training from SEI puts more emphasis on the product improvement aspects of these techniques rather than the personal process improvement that the analysis of their data should motivate.³

The Team Software Process is now under development at SEI to apply PSP principles to small teams and bridge the gap between PSP and CMM practices. Formal inspections feature preeminently in TSP activities. The “Quality Manager” and “Process Manager” (two of the major role of TSP) cooperate to lead the team in improving the process based on the analysis of the results of these inspections.

This paper presents original data gathered from nearly 100 engineers in diverse industries to assess the effectiveness of PSP reviews. It discusses the limitations of these techniques and contrasts the data with published results from formal inspections. Some limited data from early TSP experiments in India are also presented.

Finally, a multi-stage review model is presented to characterize the synergy between PSP reviews, team reviews, and formal inspections. Observations, lessons learned and quantitative tidbits of information will be offered during the talk in the spirit of the experience factory.

³ PSP students do analyze their data in mid term report R4 and final report R5 and produce Process Improvement Proposals along the way for nine of the ten programs they write during the class.

2. PSP reviews

Figure 2 shows the evolution of the “process yield” during a PSP class recently given at a major software company in New Jersey. The PSP “Process Yield” is defined as the number of defects removed before first compile divided by the number of defects injected before compile and expressed in percent. The numbers on the X axis represent the programs that PSP students have to produce during the 10 days training class to practice increasingly sophisticated level of their personal process.

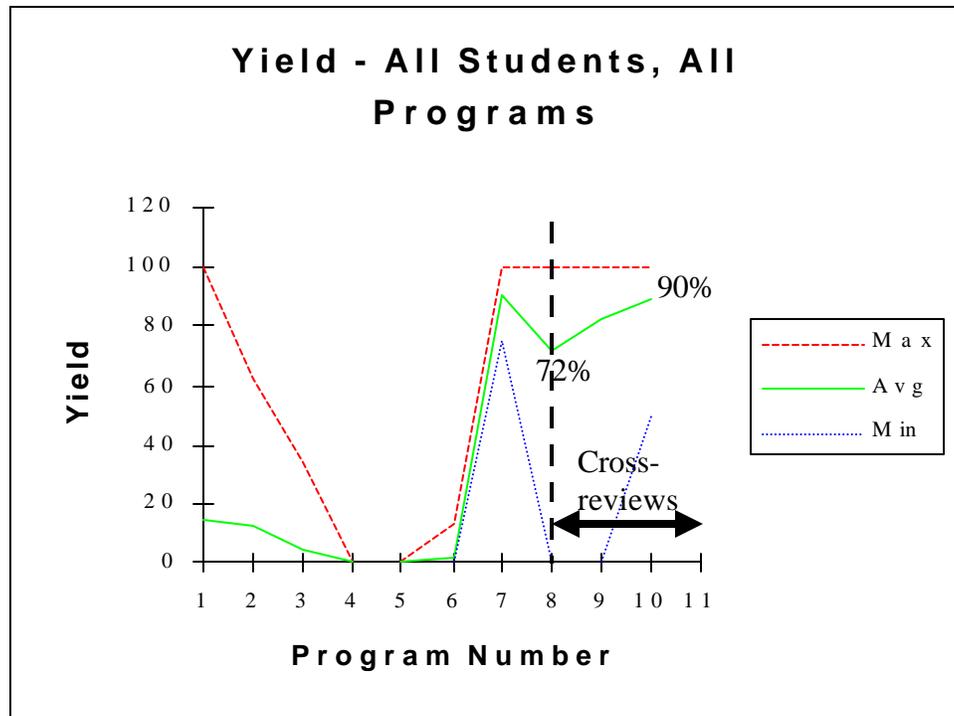


Figure 2: Improving personal review effectiveness

Before the introduction of formal personal design and code review techniques with program 7, engineers remove only a small fraction of their defects before compile. After defining their own checklists, based on the analysis of their own defects, the average yield rises to reach 90% by the end of the class. This result will not surprise anyone familiar with Cleanroom techniques.

As a way to share lessons learned and exploit cognitive dissonance [Weinberg-71], the author also demands that cross reviews be held after a personal baseline has been established with program 7. This extra step explains the superior (compared to the SEI data base [Hayes-97]) results observed. Results and their statistical validity are discussed in detail during the talk.

TSP reviews

One of the metrics known to have an impact on yield is the number of lines of code reviewed per hour. Gilb advises to not exceed 200 executable LOC reviewed per hour to maximize the chance of finding defects during inspections [Gilb-93]. The PSP class uses the same number.

Figure 3 shows an attempt at finding an explicit negative linear regression relationship between yield and speed with PSP class data. Tremendous variation in individual performance results in a very poor correlation ($R^2 < 0.2$). It must be noted that this data was gathered in a small group (10 students) featuring a wide variation in individual experience and programming languages.

Defects removed by CR

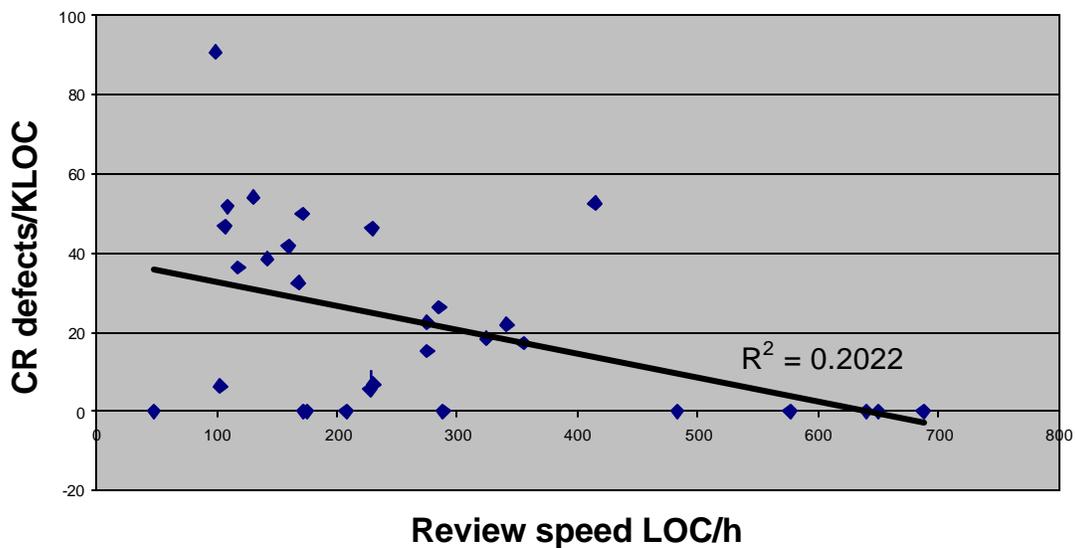


Figure 3: Poor correlation between code review speed and yield

With Team software process (Fagan style) inspections, correlation is not that much better but limited author's team data obtained through TSP0.3 experiments in India shows a clearer difference in yield between "fast reviews" (faster than 250 LOC/h) and the rest (Figure 4).

This time the group is much more homogeneous, the individuals having very comparable experience, using the same programming language and environment in a production setting and, above all, having completed PSP training and TSP launch as a team.

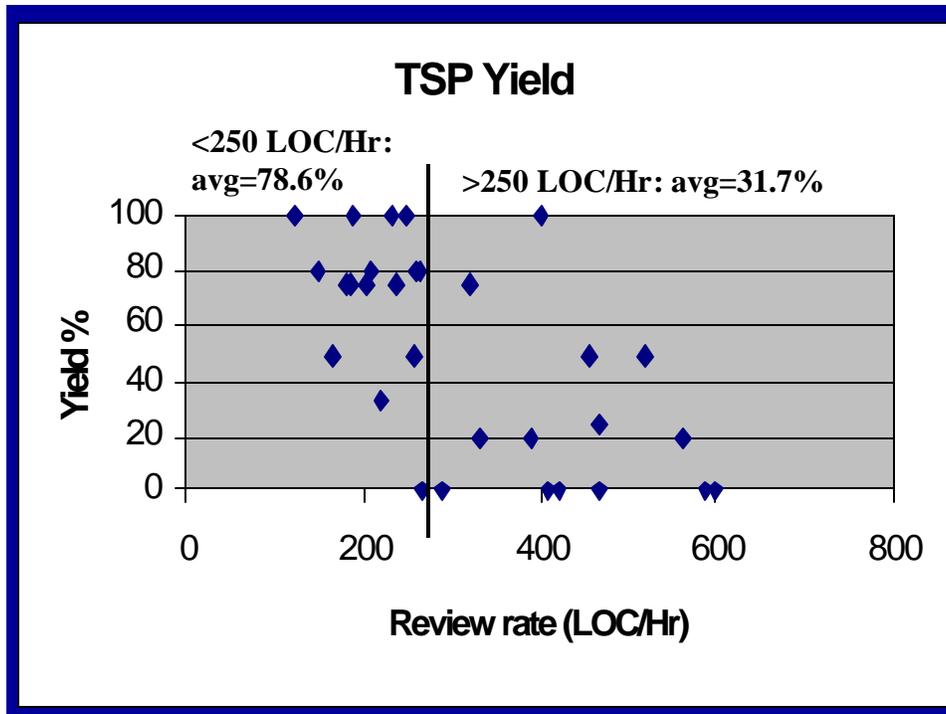


Figure 4: Respecting the TSP speed limit

3. Fagan's inspections

Barry Boehm in his landmark work of over 20 years ago provided highly practical information on the cost of defects found at various stages of the process [Boehm-82]. In a recent study of DoD contractors, Don O'Neill provided similar data accompanied with insightful comments showing the benefit of Fagan class inspections [O'Neill-98]. Their data is summarized in Table 1 below.

Phase removed	Time to fix defect
Code inspection	30 minutes
Integration testing	2-10 hours
System testing	10-40 hours

Table 1: Cost of defects

If a model could be built to characterize defect removal effectiveness and costs across the process, the impact of review techniques at each phase of the life cycle could be quantitatively studied and their return on investment precisely determined. The following paragraphs describe one step toward such a model.

4. A data driven model

Figure 5 represents any phase of a software process as a filter, actually an active filter in the sense of electronics. Noise (defect) is inherited from previous stages, some is created in the current stage, some is removed and some escape to the next stage.

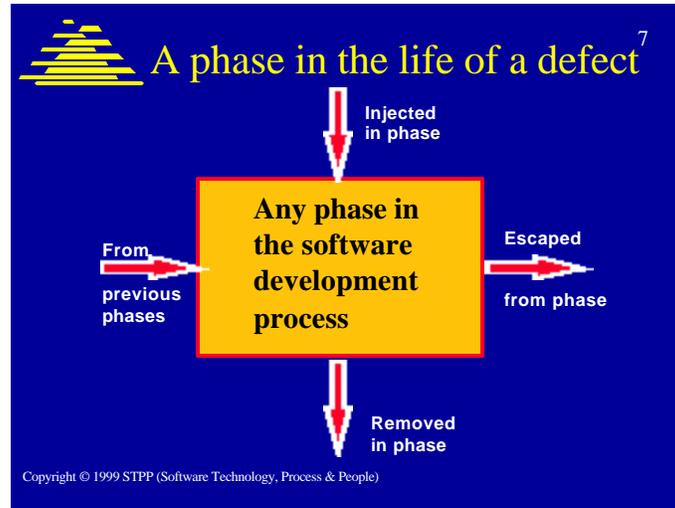


Figure 5 : Filtering defects

The goal of any software process is to maximize the signal/noise ratio by reducing the injection of defects while boosting the effectiveness of their removal (filtering). The model shows that this is best achieved at the individual level (where the defects are created) through personal review and by the application of a synergy of review techniques (cross review and inspections) to catch what was missed because of individual bias.

Figure 6 shows how several stages of filtering can help improve the quality of the software product. Here, yield is defined for each stage as the percentage of the defects removed in the stage that were present at stage entry. For instance, the first stage of yield Y_1 removes a portion of the I_1 defects present at its entry but allows $I_1(1-Y_1)$ defects to escape to later phases.

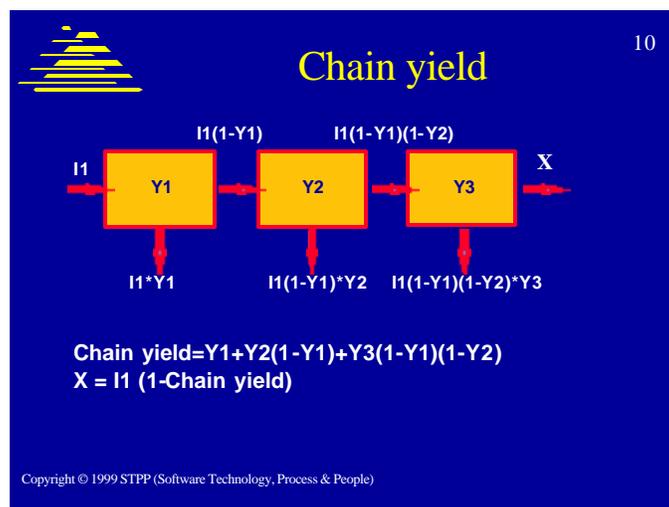


Figure 6: Cascading the filters

When several stages of filtering are combined, the contribution of each stage can be easily studied and a composite (akin to a transfer function in electronics), or chain yield can easily be computed and, therefore the number X of defects left in the product after all the reviews can be predicted.

The chain yield is clearly more dependent on the yield of the early stages since the multiplication factors (1-Y) are all lower than one. Figure 7 shows the impact of boosting the first yield from 50% to 80% on the defect escaping a chain of reviews everything else (such as other stage yields) been equal.

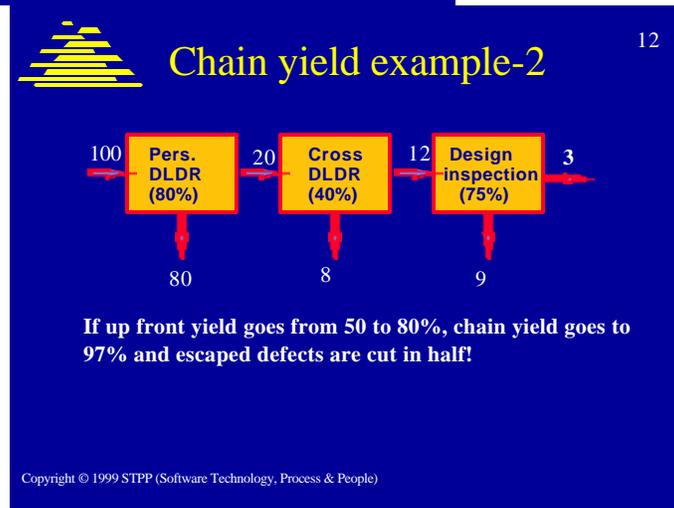
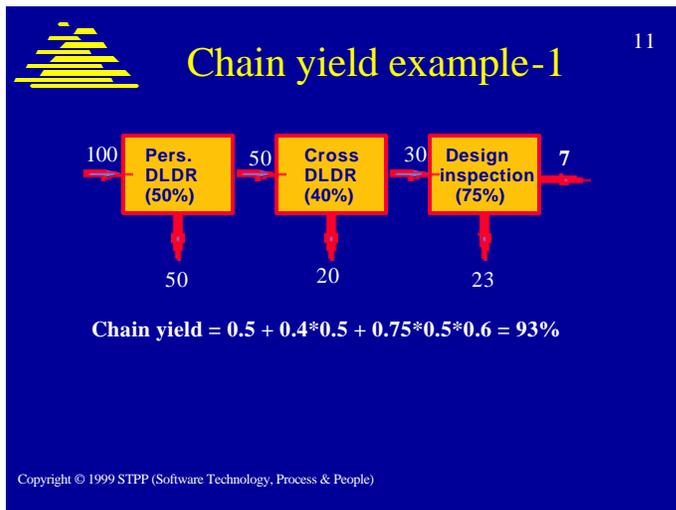


Figure 7: Impact of the earliest yield

This model can be used to study the impact of review techniques on the process overall defect removal effectiveness. It seems to show however, that boosting the yield of the individual review has the most dramatic impact on the quality of the product.

Several factors impacting individual yield, cross review yield and formal inspection effectiveness, including psychological factors, will be discussed in the talk.

5. Conclusion

The quality of the software product depends on the effectiveness of all defect detection and removal techniques used at each stage of the software process. This paper shows the importance of personal review techniques. However, the discussion will show that PSP style reviews are insufficient in a production (team-based) environment.

A simple model allows the quantitative study of the synergy between personal reviews, cross reviews and formal inspection using data gathered from the field. The next step will be to add defect detection and removal costs at each phase of the life cycle to obtain a better picture of the costs and benefits achieved by each technique and better optimize the software development process.

6. Bibliography

[Basili-89] Victor Basili, 'Software Development: A Paradigm for the Future', Proceedings of the thirteenth Annual International Computer Software & Applications Conference, Orlando, FL, September 20-22, 1989.

[Boehm-82] Barry Boehm, "Software Engineering Economics", 1982

[Gilb-93] Tom Gilb and Dorothy Graham, "Software Inspection", Addison Wesley, 1993

[Hayes-97] Will Hayes and James W. Over, "The Personal Software Process (PSP): An Empirical Study of the Impact of PSP on Individual Engineers", CMU/SEI-97-TR-001, CMU, 1997.

[Herbsleb-94] Jim Herbsleb et al, "Benefits of CMM Based Software Process Improvement", CMU/SEI-94-TR-013, August 1994.

[Humphrey-95] Watts S. Humphrey, 'A Discipline for Software Engineering', Addison Wesley, 1995.

[O'Neill-98] Don O'Neill, "National Software Quality Experiment", SEL workshop, 1998

[Weinberg -71] Gerald M. Weinberg, 'The Psychology of Computer Programming', Van Nostrand Reinhold, 1971