

PSP Metrics in Support of Software Engineering Education

Thomas B. Hilburn
Department of Computing and Mathematics
Embry-Riddle Aeronautical University
Daytona Beach, FL 32114
hilburn@db.erau.edu

Abstract

This paper describes a position about use of the Personal Software Process (PSP) metrics that was presented in the workshop: "Software Metrics: Views from Education, Research, and Training". The position presented here, describes how and why PSP metrics can be used in teaching and learning about software engineering.

1. Introduction

The Personal Software Process (PSP) is designed to help students and professional software engineers to organize and plan their work, track their performance, manage software quality, and analyze and improve their personal process [1]. The PSP can provide students of the software engineering discipline with a framework for measuring and analyzing their development work so that they produce programs of higher quality and in a more predictable manner. The author believes that when PSP is included in a computing curriculum (computer science, computer engineering, information systems, software engineering, etc.), it can have a significant effect on improving the software practices of students; and graduates of such a curriculum will be better prepared to enter the field of software engineering.

2. PSP Metrics and Their Use

Students in software engineering can be significantly improved if they collect and analyze data on the way they develop software. Students using the PSP collect size, effort and quality data on the software they produce. When PSP historical data is collected and maintained, students are able to use this data to make accurate estimates about future work. They can use historical size information to estimate the size of units and module; and they can use historical data to analyze the relationship between size and effort and then use this analysis to estimate the effort required to complete a project. Students can also use the defect data they collect to assess the quality of the software products they produce and they can evaluate the process used to develop the products.

Table 1 describes some of the basic metrics used in the PSP. In 1997 the Software Engineering Institute (SEI), conducted a study of the training effects of PSP [2]. It used the PSP data from about 3000 programs that were written by about 300 students and engineers. The study showed that the PSP can significantly improve the ability to estimate program size and effort (20%-30% improvement); it showed a major increase in the quality of the product produced (test defects were reduced from 40 defects/KLOC to 10 defects/KLOC); and it showed a positive effect on planning for quality (as the A/FR increased to 2 and above, the yield improved to between 70% and 100%).

Table 1: PSP Metrics

	Metric Type	Description
Product Quality	Total defects/KLOC	the number of defects found in development, per 1000 lines of code
	Test defects/KLOC	the number of defects found in test, per 1000 lines of code
Process Quality	Yield	the percent of defects found before compile
	Appraisal COQ (Cost of Quality)	the percent of total development time spent in design review and code review
	Failure COQ	the percent of total development time spent in compile and test
	Total COQ	Appraisal COQ + Failure COQ
	A/F R	$\frac{\text{Appraisal COQ}}{\text{Failure COQ}}$
	Review rate - LOC/hour	the number of lines of code reviewed per hour in a review (code review or design review)
	Defect removal rate - defects/hour	the rate at which defects are removed in a defect removal phase (design review, code review, compile, test)

Also, when students are following a defined software process, like the PSP, a teacher has more knowledge about how students go about their work and this offers increased opportunities for providing timely and worthwhile guidance. The teacher not only has the opportunity, but he/she has the student data to help in assessing student progress, and it forms a basis for meaningful advice. Students can be advised about the viability of their plans, about the results of reviews and tests, and about what they can do to improve their processes. The PSP allows the teacher to move into the role of a “coach”, rather than simply acting as a lecturer and “debugger of last resort”. Metrics such as the yield, the AF/R, the code review and design review rates, and the various defect removal rates allow a teacher/coach to see just how effective the student's quality efforts have been and to discern specific, detailed advice on what actions to take to improve software quality.

3. Conclusion

The PSP can not solve all problems that students and professionals have in developing software, but it can support and guide such software developers in establishing disciplined practices that can be analyzed and improved. The data and metrics provided by the PSP form the basis for an engineering and scientific approach to such an analysis, and they appear to provide more promise for success than any other method or tool on the horizon. These metrics also encourage and support a new, more effective paradigm for education that replaces the programming teacher with a coach that is able to give detailed, specific counsel on how students can improve.

4. References

- [1] Humphrey, Watts S. *A Discipline for Software Engineering*. Reading, Mass.: Addison Wesley, 1995.
- [2] Hayes, W. & Over, J.W., *The Personal Software Process (PSP): An Empirical Study of the Impact of PSP on Individual Engineers* (CMU/SEI-97-TR-001) Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1997.