Software – A performing science?

Watts S. Humphrey

The Software Engineering Institute, Carnegie Mellon University, 4500 Fifth Avenue,
Pittsburgh, PA 15213, USA
E-mail: watts@sei.cmu.edu

Technical issues are important for software work, but so are personal disciplines, team-working skills, and application-domain knowledge. Also, much like an artistic performance, first-class software engineering requires constant practice, good technique, and effective coaching. The challenge of producing high-quality large-scale software products is substantial today and will be even more demanding in the future. Without concerted action, we cannot expect software organizations’ capabilities to improve. To address these problems, the Software Engineering Institute (SEI) has developed the Personal Software Process (PSP) and the Team Software Process (TSP). This paper addresses the problems of software engineering and discusses the intellectual nature of software work. It then reviews the characteristics of this kind of work and describes the principal conditions for effective software performance. In the conclusion, the author makes some observations about the challenges ahead and the future actions required.

0. Introduction

Technical issues are important for software work, but so are personal disciplines, team-working skills, and application-domain knowledge. Also, much like an artistic performance, first-class software engineering requires constant practice, good technique, and effective coaching. However, just like a performing art, even when we know all the required steps, it is still hard to get everything right. We do not just need proper education and training, we need a supportive working environment and a management system that is more attuned to motivation and coaching than to assigning and directing.

The challenge of producing high-quality large-scale software products is substantial today and will be even more demanding in the future. While operable software is being produced with current methods, the results are far from satisfactory. The Standish group reported that, of 175,000 software projects, only 16% were successful, 31% failed, and 53% were considered “challenged.” Of these challenged projects, only 7% delivered all the expected functions and 48% had more than a 100% schedule overrun [Johnson 1995]. What is more, compared to 10 years before, executives in these organizations generally considered the situation to be deteriorating: 69% said the situation was the same or worse and 17% said the situation was significantly worse.
Only 8% felt it was substantially better. More recently, another study reported that software work has roughly the same problems in Europe and in Asia [Keil 1998].

Without concerted action, we cannot expect software organizations' capabilities to improve. What is more, in the future, we can expect that the challenges facing software groups will be much more demanding than they are today. To address these problems, we need to look for and address the root causes of the problems.

This is the approach followed by the Software Engineering Institute (SEI). We first introduced the Capability Maturity Model (CMM)\(^1\) to guide management in improving their software organizations [Paulk \textit{et al.} 1995]. Then we developed the Personal Software Process (PSP)\(^2\) to show engineers how to use a measured and defined process for high-quality software work [Humphrey 1995, 1997]. Now, with the Team Software Process (TSP)\(^3\), we are showing teams how to consistently use disciplined software methods [Humphrey 1998; Webb and Humphrey 1999; Humphrey 2000].

In this paper, I first address the problems of software engineering and discuss the intellectual nature of software work. I then review some of the characteristics of this kind of work and describe, with examples, the principal conditions for effective software performance. In conclusion, I make some observations about the challenges ahead and the future actions required.

\section{The nature of software work}

First-class software engineering is not just a problem of better training and enlightened management; it requires personal discipline. It also requires the kind of support generally needed by other demanding human activities. However, just as in other professional fields, we cannot expect our managers and customers to be knowledgeable about software methods. We need software professionals who know their own capabilities and can demand the resources and support they need for quality work. Even more importantly, we need managers who will respect and support their software engineers.

In the SEI’s work with software teams, we have found that, with modest training and experience and with proper management and support, software professionals can do extraordinary work. They can regularly deliver quality products on their committed schedules and for less than the planned costs. When they work this way, engineers are more productive and they have more fun. Software professionals are potentially capable of producing unbelievable results, but we need to better utilize their talents.

\(^1\) Registered in the U.S. Patent and Trademark Office.
\(^2\) Personal Software Process and PSP are service marks of Carnegie Mellon University.
\(^3\) SM Team Software Process and TSP are service marks of Carnegie Mellon University.