Books

Measuring Software Reuse: Principles, Practices, and Economic Models, Jeffrey S. Poulin, Addison Wesley Longman, Inc., Reading, MA, 1997. 195 pp. (ISBN 0-201-63413-9).

In any engineering discipline, the maturity of its measurement process is a pretty good indicator of the maturity of the underlying theory and practice. Measurement usually begins simply, with categories that distinguish something as having a property from something else that does not. For example, early measurement of temperature probably involved someone's saying "this is too hot" and "this is not too hot" for some purpose. Measurement helps us to form theories and understand principles, so that categories such as "not too hot" soon become ordered descriptions, such as "too hot," "very hot," "hot," and "not hot enough." Eventually, we develop a sophisticated understanding, and along with it, more sophisticated ways of measuring important attributes of the items of interest; measuring the length of a column of mercury, and converting from Celsius to Fahrenheit scale, are far cries from poking your hand into water to determine if it is too hot.

For this reason, Poulin's *Measuring Software Reuse* is an important step toward a clearer understanding of how software reuse works, and how we can make it work better. These days, most managers' lists of best practices include software reuse, but there is only anecdotal evidence about the best ways to ensure that reuse is practical and effective. Poulin reviews the reuse literature and begins to synthesize the metrics and models that are most appropriate in measuring the extent of reuse in an organization, as well as the effects of reuse on a company's bottom line.

Poulin is quite realistic, managing our expectations from the very beginning. He notes right away that there are many different kinds of reuse, from incorporating a component unchanged to modifying a component in various ways. He also explains that code is not the only development artifact that can be reused; we can also use requirements, designs, documentation, test scripts, and test data from already-completed projects. However, this broader reuse is less well-understood than code reuse, so Poulin restricts his measurements to black-box code reuse: using code developed for another project, and using it without any modification whatsoever. He takes great care in explaining what kinds of code are counted as reuse; databases and commercial off-the-shelf products are not considered to be reuse, but borrowing code from another project or organization is. Poulin also justifies measurement in terms of lines of code, suggesting that it is not perfect, but it is the best we can do for now.

Many managers do not distinguish reuse from reusability, but Poulin sets them straight by defining the relative cost of reuse (RCR) and the relative cost of writing for reuse (RCWR). RCR reflects the amount of effort saved by using code that is already written, while RCWR describes the extra effort required to make a new product reusable for future projects. Then, he reviews the literature and his experiences at IBM and Lockheed Martin Federal Systems to suggest guidelines for RCR and RCWR when historical data are not yet available for your organization. Poulin also discusses related measures and models, reviews many (but not all) of the published reuse-related cost models, and suggests how existing measures and models might apply to your company's or organization's situation. He goes on to discuss measures related to reusability (how do you tell when a module is a good candidate for reuse on another project?) and repositories (what are the costs associated with categorizing and storing reusable modules for future use?). In these discussions, he points out the real difficulties in capturing the right kinds of measurements. Attempts to restrict measurements to easy-to-measure, structural metrics have not yielded satisfactory predictors of reusability; likewise, attempts to determine when extracted components are actually used must rely on the conscientious user's supplying information after the fact.

0018-8670/97/$5.00 © 1997 IBM

But Poulin points out that there are some simple things we can measure as we learn more about reuse, reusability, and more sophisticated measurements.

Claims in support of reuse involve higher quality as well as decreased effort to code and integrate components. Poulin spends less time on quality measurements than on effort, but this is probably because there is far less information available about reuse quality in the literature. However, there is a key aspect of quality that should be discussed, regardless of its measurement maturity: who is responsible for the quality of a reused component? Is the person who places the component in the repository, or is it the reuser? Similarly, we need to know exactly how reuse activities should be incorporated in the larger development process, so that developers know exactly where reuse fits in their daily work. These questions and issues have significant implications for configuration management, repository management, and the incentives used for encouraging reuse in an organization. Indeed, the success of a reuse program may depend far more on the answers to these and similar questions than on good measurements of reuse level and reusability. Poulin gives us guidelines about what and how to measure, in a very restricted sense of reuse. But he does not provide guidelines about how to make reuse part of an organization's culture; many of these cultural issues are not related to measurement, but they are clearly related to the success and maturity of the practice.

The lack of a broader view is not by any means a defect; Poulin never suggests that his book will provide everything you need to know about reuse. Indeed, he raises many important questions, some of which require further understanding of the reuse process. This interaction between measurement and process is typical and healthy; the back-and-forthing between what to measure, how to measure, and what it all means in a larger process is the way we learn and mature. My only quibble with Poulin's approach is that he suggests (and states explicitly) that we do not need any more models of reuse. In my experience, the most useful models are the ones that are tailored to particular organizations and cultures; Poulin admits as much when he describes the boundary problem (that is, where are the reused components coming from and going to?). It is essential that we apply his ideas to our own situations, using historical data that reflects our organizational cultures and goals. As in any engineering discipline, our models and measurements will help us to mature and be more effective. The fact that Poulin's book focuses on a substantial body of reuse measurement means that we are well on our way to understanding and perfecting reuse, to broadening it to include all development artifacts, and to making it a natural way of doing business.

Shari Lawrence Pfleeger
Systems/Software, Inc.
Washington, D.C.

Managing Technical People: Innovation, Teamwork, and the Software Process, Watts S. Humphrey, Addison Wesley Longman, Inc., Reading, MA, 1997. 326 pp. (ISBN 0-201-54597-7).

Managing Technical People is the kind of book you'll want to take along on your career. Whether you are dealing with front-line management challenges, are firmly on your way through middle management, or are setting policy for the entire department, Watts S. Humphrey provides proven perspectives and approaches you can apply to your unique situation. Even if you have chosen a technical career path rather than management, there is much in this book for you.

There is no sense of a glib "two-minute manager" in *Managing Technical People.* Nor will you hear the orchestra tuning up for a rendition of *How to Succeed in Business Without Really Trying.* Instead, with the idea that history is a marvelous teacher as long as we are willing to learn, Humphrey manages to give a comprehensive view of technical organizations and those who must manage them.

Humphrey divides *Managing Technical People* into eight complementary, internally cohesive parts. For example, Part One concentrates on the manager as a leader. Beginning with the idea that "the technical leader's most important role is to set goals and drive unswervingly to meet them," Humphrey identifies leadership styles and stresses the importance of commitment. In the process, he not only identifies elements of responsible commitments, but analyzes overcommitment and crusades, as well as the tricky and potentially explosive issue of changing commitments. Humphrey then turns to the elements of professionalism and makes specific suggestions on how managers can encourage professionalism in their technical staff. Part One concludes with a chapter entitled "Respect for Individuals." Humphrey sug-

gests that respect for the individual rests on an attitude of fairness, and provides four criteria to evaluate how respectful a work environment is.

The remaining parts, "Managing Technical and Professional People," "The Identification and Development of Talented People," "Innovation," "Innovative Teams," "The Organization," "Managing Change," and "Strategies for Managing Change," reflect similar internal cohesiveness. Humphrey fills each chapter with plenty of appropriate examples and checklists so readers can see how their workplace measures up.

Even those technical professionals with no interest in a management career should study Humphrey's sections on recognizing and developing technical talent. His chapter entitled "Identifying Talented People" lists criteria valuable to those serious about gaining the right kind of visibility. So, too, technical aspirants can compare the events of their own careers against Humphrey's coaching on how to develop the ". . . high potential technical professionals."

Still, *Managing Technical People* is, true to its title, a book focused on management techniques and perspective. All can benefit from understanding the context for organizational growth, managing change, political dynamics, and power relationships that Humphrey describes.

The senior executive can effectively apply the book's organization maturity principles to Maslow's human needs hierarchy or Deming's quality management model. Such executives may find Humphrey's counsel on remedial structure changes appropriate to their situation, or recognize their own organization when the book describes the unique problems of an aging organization.

The middle manager benefits from Humphrey's treatment of the innovation process—particularly as he identifies the roles of "inventor," "champion," and "sponsor." Such managers may be able to introduce some form of the book's suggestions on organizational integration and support systems into their departments, and they may want to couple those systems with an awards and recognition program using guidelines supplied in the book.

The front-line manager can immediately apply the book's insights on the dynamics of technical teams. Such managers will learn how to energize their project areas with leadership, commitment, development, and motivation focused on the unique nature of the technical workplace environment.

Though Humphrey manages to remain impartial for most of his book, he tends to lose objectivity when sharing the organizational models from Carnegie Mellon University. This is understandable, of course, in that Humphrey is a full Fellow at the Software Engineering Institute (S.E.I.) of Carnegie Mellon University, where he founded and led S.E.I.'s process program. Humphrey both consults and lectures on his books and the Institute's organizational models. Still, the strategies he describes are valuable, interesting, and worthwhile.

If you only read one book this year, or are looking for a universal management formula, look elsewhere. But if you seek a resource that provides insights, evaluation tools, and counsel you can use now and throughout your working career, *Managing Technical People* by Watts S. Humphrey is hard to beat.

Brian Case
Lisle
Illinois

Note—The books reviewed are those the Editor thinks might be of interest to our readers. The reviews express the opinions of the reviewers.