



Managing Defects Together



Many software managers and practitioners consider peer reviews to be a principal defect management technique. The purpose of disciplined peer reviews is to find and remove defects early, which in turn will reduce rework later. Peer reviews can take place during all phases of a software project life cycle and can be a review of any work product. Planning documents, schedules, requirements specifications, interface documents, test plans, and software code are examples of work products that can be peer reviewed. Although this issue of *CrossTalk* is far from a piece of code, it is a work product and has been subject to a formal review process similar to a peer review.

For those of you who have never had the opportunity to participate in a disciplined peer review, here is some insight from my peer review experiences while on a software development team. In most cases, I was given paperwork (code) to review approximately a week ahead of the scheduled peer review meetings. I found it most comforting that the focus of the peer review meeting was on the work products and not the software developers. We used checklists to guide us in finding defects.

My roles in the peer reviews varied from leader, recorder, and reviewer to author. As a code-writer (or author), the meetings were a great help to me as work product defects were uncovered in a nonconfrontational team setting. The roles and responsibilities of the review members, as well as the meeting goals, were well understood by all. This helped make the review process routine and a time for team support. Through my peer review experiences, I gained an invaluable understanding for project teams working together to meet the end goal of producing a quality product.

As we focus this month's issue on defect management, we begin with *The Bug Life Cycle* by Lisa Anderson and Brenda Francis. This article emphasizes the need for teams to carefully handle and track software bugs throughout a project's life cycle. Defect tracking through a database and enforcing policies and procedures are some of the methods suggested to improve testing and quality assurance efforts.

Next, two approaches to acquiring and developing quality software are compared by Dr. Kenneth D. Shere in his article *Comparing Lean Six Sigma to the Capability Maturity Model*. Shere discusses the primary differences and common goals between these process improvement approaches aimed at reducing software defects. Our issue continues with a look at defect management when utilizing object-oriented approaches. In *Managing Software Defects in an Object-Oriented Environment*, Houman Younessi presents a fault model and describes the many steps and opportunities to detect and remove defects in an object-oriented environment.

The Software Engineering Institute's Personal Software ProcessSM (PSPSM) is yet another disciplined method aimed at managing and reducing defects. Iraj Hirmanpour and Joe Schofield share their experience with the PSP defect management framework in *Defect Management Through the Personal Software Process*. They provide good insight into how the use of defect collection and analysis metrics can benefit organizations engaged in software process improvement.

In *Defect Management in an Agile Development Environment* by Don Opperthausen, we are reminded of the importance of finding defects in the requirements phase of software projects. Opperthausen discusses a set of best practices associated with the Agile+ methodology that focuses on the prevention of requirements and implementation defects. In *Lessons Learned From Another Failed Software Contract*, Dr. Randall W. Jensen's examination of a failed major avionics modernization program uncovers issues that continue to derail software development projects.

In our Open Forum section this month, Raymond Grossman brings us *Defect Management: A Study in Contradictions*. Grossman discusses how the utility of defects can be diminished throughout the software development process if management is not involved and supportive of defect reporting and documentation. Also in this issue, we are once again proud to announce the third annual U.S. Government's Top 5 Quality Software Projects contest. You can submit your 2003 nomination at <www.stsc.hill.af.mil> then select the *CrossTalk* site and click on the Top 5 option.

Special thanks to all of our authors this month for sharing their lessons learned and best practices regarding software quality and defect management. I hope you find this month's issue helpful as you and your teams strive to learn more about handling and managing software defects to meet the end goal of high quality work products for your customers.

Tracy L. Stauder
Publisher