Evolution of Quality in the Software CMM

Mark C. Paulk
Software Engineering Institute
Carnegie Mellon University

The Process Management Principle – The quality of a product is largely governed by the quality of the process used to build it

The fundamental principle underlying the Software Engineering Institute’s process work is the process management principle, but not everyone using the Capability Maturity Model℠ (CMM℠) for Software has a good understanding of what is meant by "quality." Quality is not an absolute concept – quality is determined by "business" context.

What does "quality" mean in your business? Is it defined as defect density? Mean-time-to-failure? Time-to-market? Customer satisfaction? Market share and profitability? Whose opinion of "quality" really counts when making decisions? Marketing? The external customer? The end user? If customer satisfaction is important, can you identify who the customer is?

I hope that we all agree that quality is more than the absence of defects. Quality can be defined from many stakeholder perspectives: the business, customers, employees, shareholders, executives – even those not using the organization’s products, but who are affected by them, e.g., environmental concerns. Quality is frequently defined as defect density, but from the user’s perspective, reliability is a much more important quality attribute than the number of defects. Defect density and reliability are only weakly related. In one study, about 35% of the software defects had a mean-time-to-failure of over 5,000 years, while only 2% had an MTTF of less than 5 years. In contractual software development and maintenance, predictable costs and schedules are important to the customer.

David Garvin identified nine dimensions of quality in a 1987 Harvard Business Review article: performance, features, functionality, safety, conformance, reliability, durability, service, and aesthetics. Other authors identify many more, but this indicates the many facets of "quality."

Although not explicitly stated in Software CMM version 1.1, the definition of quality evolves as an organization moves up the maturity levels. At Level 2 the emphasis in Requirements Management is on conformance to requirements. As Crosby suggests, if you do not build what the customer has specified, you will not have a quality product. At Level 3, one of the goals of Intergroup Coordination is that "the customer’s requirements are agreed to by all affected groups." This includes the customer and end users – in Activity 1, the engineering groups, customer, and end users work together to establish the system requirements. This is a more proactive statement than Level 2’s focus on understanding the requirements. It implies a collaborative approach, with the result that quality is defined by the customer, as Deming states, in a more meaningful way. At Level 4, we can characterize Software Quality Management as addressing Juran's fitness for use in a quantitative sense. Management by fact, using tools such as Quality Function Deployment, leads to a profound knowledge of the customer, the application domain, and the tradeoffs involved in building a quality product.

In one sense the evolving nature of quality is a subtle point. The real point is that organizations undergo a culture shift as they mature. The view of quality – and the customer – evolve as organization builds its process capability, even if the shift is not a conscious one.

Level 4 organizations also address one issue that, while not explicitly stated, is crucial to quality management: the problem of suboptimization. Software Quality Management deals with project-level quality issues from a systems perspective. Quantitative Process Management deals with process control at an activity level, with individual professionals and teams controlling quality on a day-to-day basis using the

※ CMM is a registered trademark of Carnegie Mellon University.
insights provided by quantitative and statistical tools. To prevent suboptimization in individual processes implies the systems-level perspective of Software Quality Management is being applied to resolve conflicting quality goals as realized in Quantitative Process Management – a tight coupling of concerns from very different perspectives!

Part of the cultural shift that occurs as an organization matures its software processes is an increasingly profound knowledge of what quality truly means – to the organization, to its customers, and to the end users of our products. The first step for most organizations is simply to build what the customer says they want, and then expand on that simple, but critical capability, by probing needs and then making informed trade-off decisions in cooperation with the customer. This is a difficult journey for most software organizations, but a necessary one for success.