Editor in Chief: Steve McConnell ■ Construx Software ■ software@construx.com

# Real Quality For Real Engineers

Steve McConnell

For decades, experts have struggled to define quality. Edwards Deming said that the only definition of quality that mattered was the consumer's.[1] Joseph Juran said that quality was fitness for use.[2] Philip Crosby provided the strictest definition of quality as "conformance to requirements."[3]

## Conformance to requirements

Although they differ on the details, quality experts agree that the customer's view of requirements is critically important. For that reason, I've found Crosby's definition of "conformance to requirements" to be the most useful definition in examining software quality. Taking into account many software projects' tendency to elicit some but not all of the customer's complete requirements, "requirements" cannot be interpreted solely as the written requirements. Requirements must also include implicit requirements—those that the customer assumes regardless of whether the development team happens to write them down. Thus, the working definition of quality that I use is "conformance to requirements, both stated and implied."

## The "ities" of software quality

In addition to specific functional requirements, software quality is also affected by common nonfunctional characteristics that are often referred to as the "ities." The ities that affect software's internal quality (quality visible to the software's developers) include maintainability, flexibility, portability, reusability, readability, scalability, testability, and understandability. The ities that affect the software's external quality (visible to the customer) include usability, reliability, adaptability, and integrity, as well as correctness, accuracy, efficiency, and robustness.[4]

Some of these characteristics overlap, but all have different shades of meaning that are desired more in some cases and less in others. The attempt to maximize certain characteristics invariably conflicts with the attempt to maximize others. Figure 1 presents a summary of the ways in which external quality characteristics affect each other.

These characteristics will be prioritized differently on different projects, which means the software quality target is always changing. Finding an optimal solution from a set of competing, changing objectives is challenging, but that's part of what makes software development a true engineering discipline.

## From product quality to project quality

When software people refer to quality, we usually refer to the quality of the software *product* we are producing. From a management perspective, however, customers also have requirements for *projects*. I think it's reasonable to draw an analogy from products to projects, conceiving *project quality* as conformance to requirements, both stated and implied. Customers' functional requirements for projects draw from a small number of possible attributes, namely schedule, resources, cost, and quality of the product

IEEE Software

**DEPARTMENT EDITORS**

**Bookshelf**: Warren Keuffel, wkeuffel@computer.org

**Construction**: Andy Hunt and Dave Thomas, Pragmatic Programmers, {Andy, Dave}@pragmaticprogrammer.com

**Country Report**: Deependra Moitra, Lucent Technologies d.moitra@computer.org

**Design**: Martin Fowler, ThoughtWorks, fowler@acm.org

**Loyal Opposition**: Robert Glass, Computing Trends, rglass@indiana.edu

**Manager**: Don Reifer, Reifer Consultants, dreifer@sprintmail.com

**Quality Time**: Jeffrey Voas, Cigital, voas@cigital.com

**STAFF**

Senior Lead Editor
**Dale C. Strok**
dstrok@computer.org

Group Managing Editor
**Crystal Chweh**

Associate Editors
**Jenny Ferrero and Dennis Taylor**

Staff Editors
**Shani Murray, Scott L. Andresen, and Kathy Clark-Fisher**

Magazine Assistants
**Dawn Craig**
software@computer.org
**Pauline Hosillos**

Art Director
**Toni Van Buskirk**

Cover Illustration
**Dirk Hagner**

Technical Illustrator
**Alex Torres**

Production Artist
**Carmen Flores-Garvey**

Executive Director
**David Hennage**

Publisher
**Angela Burgess**

Assistant Publisher
**Dick Price**

Membership/Circulation Marketing Manager
**Georgann Carter**

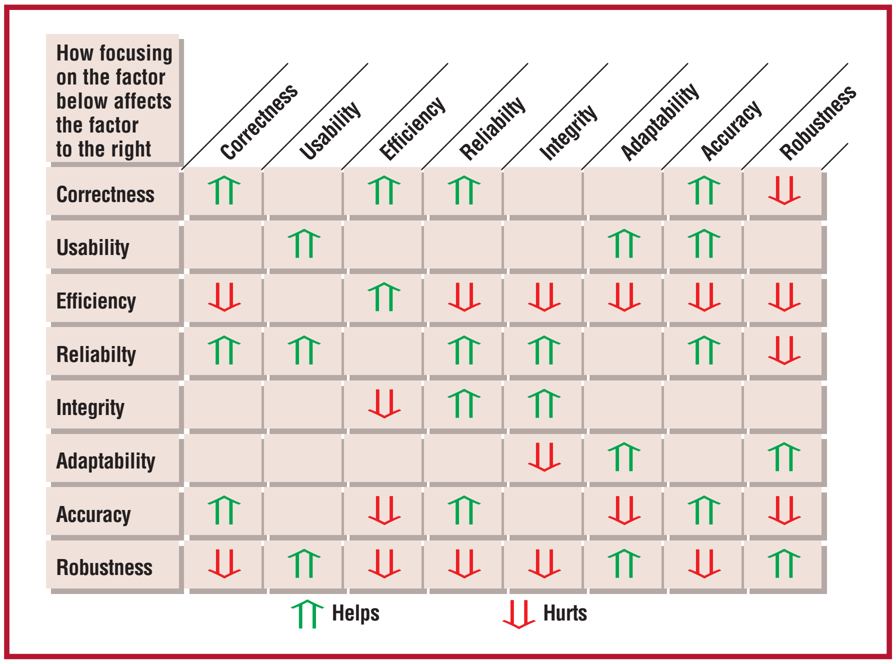
Advertising Assistant
**Debbie Sims**

**CONTRIBUTING EDITORS**

Greg Goth, Denise Hurst, Gil Shif, Keri Schreiner, and Margaret Weatherford

**Editorial:** All submissions are subject to editing for clarity, style, and space. Unless otherwise stated, bylined articles and departments, as well as product and service descriptions, reflect the author's or firm's opinion. Inclusion in *IEEE Software* does not necessarily constitute endorsement by the IEEE or the IEEE Computer Society.

**To Submit:** Send 2 electronic versions (1 word-processed and 1 postscript or PDF) of articles to Magazine Assistant, *IEEE Software*, 10662 Los Vaqueros Circle, PO Box 3014, Los Alamitos, CA 90720-1314; software@computer.org. Articles must be original and not exceed 5,400 words including figures and tables, which count for 200 words each.



| How focusing on the factor below affects the factor to the right | Correctness | Usability | Efficiency | Reliability | Integrity | Adaptability | Accuracy | Robustness |
|---|---|---|---|---|---|---|---|---|
| Correctness | ⇈ | | ⇈ | ⇈ | | | ⇈ | ⇊ |
| Usability | | ⇈ | | | | ⇈ | ⇈ | |
| Efficiency | ⇊ | | ⇈ | ⇊ | ⇊ | ⇊ | ⇊ | ⇊ |
| Reliabilty | ⇈ | ⇈ | | ⇈ | ⇈ | | ⇈ | ⇊ |
| Integrity | | | ⇊ | ⇈ | ⇈ | | | |
| Adaptability | | | | | ⇊ | ⇈ | | ⇈ |
| Accuracy | ⇈ | | ⇊ | ⇈ | | ⇊ | ⇈ | ⇊ |
| Robustness | ⇊ | ⇈ | ⇊ | ⇊ | ⇊ | ⇈ | ⇊ | ⇈ |

⇈ Helps ⇊ Hurts

**Figure 1. Interactions between product quality external characteristics.**

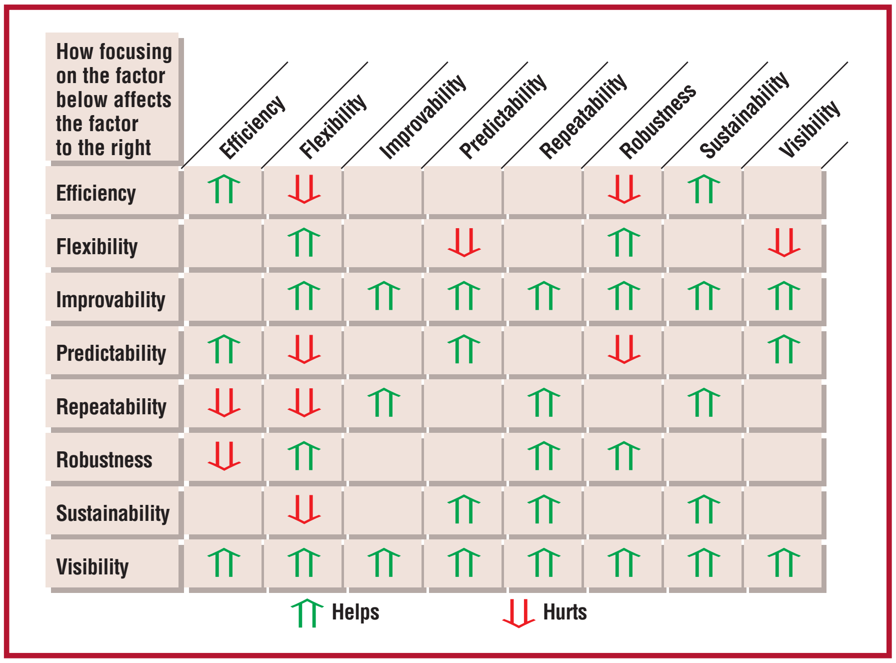
produced. In some cases, a customer might prioritize cost higher—in others, schedule or product quality.

Additionally, project quality includes nonfunctional requirements such as

- *Efficiency*: Minimal use of schedule, budget, and staff to deliver a particular software product.
- *Flexibility*: The extent to which the project can be modified to deliver software other than that for which the project was originally intended or to respond to changes in project goals.
- *Improvability*: The degree to which project experiences can be fed back into the project to improve project performance.
- *Predictability*: The degree to which a project's cost, schedule, and product quality outcomes can be forecast in advance.
- *Repeatability*: The degree to which the project after the current project can be conducted using practices similar to those used on the current project.
- *Robustness*: The degree to which the project will continue to function in the presence of stressful environmental conditions.
- *Sustainability*: The duration for which a project can continue using its current practices.
- *Visibility*: The ability of a customer to accurately determine project status and progress.

These project characteristics interplay with each other just as the software quality attributes do. Figure 2 shows the interactions. In addition to the interactions shown in Figure 2, some of these project quality characteristics tend to support or undermine the various product characteristics summarized in Figure 1.

Different projects have different priorities among efficiency, flexibility, improvability, and the other characteristics shown in Figure 2. An established business might place high values on efficiency, predictability, improvability, and repeatability. A start-up company might place a higher value on robustness and visibility; it might not value sustainability and repeatability at all. This suggests that there isn't one best definition of project quality for all projects; the best definition depends on the project's consumers and those consumers' specific project requirements.



| How focusing on the factor below affects the factor to the right | Efficiency | Flexibility | Improvability | Predictability | Repeatability | Robustness | Sustainability | Visibility |
|---|---|---|---|---|---|---|---|---|
| Efficiency | ⇈ | ⇊ | | | | ⇊ | ⇈ | |
| Flexibility | | ⇈ | | ⇊ | | ⇈ | | ⇊ |
| Improvability | | ⇈ | ⇈ | ⇈ | ⇈ | ⇈ | ⇈ | ⇈ |
| Predictability | ⇈ | ⇊ | | ⇈ | | ⇊ | | ⇈ |
| Repeatability | ⇊ | ⇊ | ⇈ | | ⇈ | | ⇈ | |
| Robustness | ⇊ | ⇈ | | | ⇈ | ⇈ | | |
| Sustainability | | ⇊ | | ⇈ | ⇈ | | ⇈ | |
| Visibility | ⇈ | ⇈ | ⇈ | ⇈ | ⇈ | ⇈ | ⇈ | ⇈ |

⇈ Helps ⇊ Hurts

**Figure 2. Interactions between project quality characteristics.**

## Real engineering

One difference between a craftsman and an engineer is that a craftsman defines quality on his own terms, whereas an engineer defines quality through his customers' eyes. The craftsman settles into a way of working that suits him personally, while the engineer adapts his approach on each project to best satisfy his customer's requirements.

Software engineering purists argue that software should always be produced to the highest level of quality, by which they mean the highest levels of product quality. End-user requirements certainly should be considered, but the organization that builds and sells the software is another consumer whose requirements must be taken into account. The product characteristics that constitute quality to the end user do not necessarily satisfy the software-developing organization's project quality requirements.

As Deming pointed out in *Out of the Crisis*, different consumers can have different definitions of quality for the same product, and this applies as much to project quality as it does to product quality. The project team, manager, and sponsoring organization can all be considered consumers of a project. A manager might consider a project to have high quality if it provides good visibility, robustness, and repeatability. The project team might value efficiency, improvability, and sustainability. The sponsoring organization might value predictability and flexibility.

A manager who factors product quality into the project plans but ignores project goals takes an abridged view of software quality. One hallmark of engineering work is the constant balancing of trade-offs. With the extensive trade-off decisions required to balance both software product attributes and software project goals, software personnel have abundant opportunities to hone their engineering skills in this area.

## References

1. W. Edwards Deming, *Out of the Crisis*, MIT Press, Cambridge, Mass., 2000.
2. J.M. Juran, *Juran's Quality Handbook*, McGraw-Hill, New York, 1998.
3. P.B. Crosby, *Quality Is Free: The Art of Making Quality Certain*, Mentor Books, Denver Colo., 1992.
4. S. McConnell, *Code Complete*, Microsoft Press, Redmond, Wash., 1993.

EDITOR IN CHIEF:
**Steve McConnell**
10662 Los Vaqueros Circle
Los Alamitos, CA 90720-1314
software@construx.com

EDITOR IN CHIEF EMERITUS:
Alan M. Davis, Omni-Vista

**ASSOCIATE EDITORS IN CHIEF**

**Design**: Maarten Boasson, Quaerendo Invenietis
boasson@quaerendo.com
**Construction**: Terry Bollinger, Mitre Corp.
terry@mitre.org
**Requirements**: Christof Ebert, Alcatel Telecom
christof.ebert@alcatel.be
**Management**: Ann Miller, University of Missouri, Rolla
millera@ece.umr.edu
**Quality**: Jeffrey Voas, Cigital
voas@cigital.com
**Experience Reports**: Wolfgang Strigel, Software Productivity Center; strigel@spc.ca