Chapter 21

Software Quality Assurance

Slide Set to accompany
Software Engineering: A Practitioner's Approach,
by Roger S. Pressman

Comment on Quality

- Phil Crosby once said:
  - The problem of quality management is not what people don't know about it. The problem is what they think they do know . . . In this regard, quality has much in common with sex.
  - *Everybody is for it.* (Under certain conditions, of course.)
  - *Everyone feels they understand it.* (Even though they wouldn't want to explain it.)
  - *Everyone thinks execution is only a matter of following natural inclinations.* (After all, we do get along somehow.)
  - *And, of course, most people feel that problems in these areas are caused by other people.* (If only they would take the time to do things right.)

Elements of SQA

- **Standards**
- **Reviews and Audits**
- **Testing**
- **Error/defect collection and analysis**
- **Change management**
- **Education**
- **Vendor management**
- **Security management**
- **Safety**
- **Risk management**

Role of the SQA Group-I

- **Prepares an SQA plan for a project.**
  - The plan identifies
    - evaluations to be performed
    - audits and reviews to be performed
    - standards that are applicable to the project
    - procedures for error reporting and tracking
    - documents to be produced by the SQA group
    - amount of feedback provided to the software project team
- **Participates in the development of the project's software process description.**
  - The SQA group reviews the process description for compliance with organizational policy, internal software standards, externally imposed standards (e.g., ISO-9001), and other parts of the software project plan.

Role of the SQA Group-II

- **Reviews software engineering activities to verify compliance with the defined software process.**
  - identifies, documents, and tracks deviations from the process and verifies that corrections have been made.
- **Audits designated software work products to verify compliance with those defined as part of the software process.**
  - reviews selected work products; identifies, documents, and tracks deviations; verifies that corrections have been made
  - periodically reports the results of its work to the project manager.
- **Ensures that deviations in software work and work products are documented and handled according to a documented procedure.**
- **Records any noncompliance and reports to senior management.**
  - Noncompliance items are tracked until they are resolved.

SQA Goals (see Figure 16.1)

- **Requirements quality.** The correctness, completeness, and consistency of the requirements model will have a strong influence on the quality of all work products that follow.
- **Design quality.** Every element of the design model should be assessed by the software team to ensure that it exhibits high quality and that the design itself conforms to requirements.
- **Code quality.** Source code and related work products (e.g., other descriptive information) must conform to local coding standards and exhibit characteristics that will facilitate maintainability.
- **Quality control effectiveness.** A software team should apply limited resources in a way that has the highest likelihood of achieving a high quality result.

Goal	Attribute	Metric
Requirement quality	Ambiguity	Number of ambiguous modifiers (e.g., many, large, human-friendly)
	Completeness	Number of TBA, TBD
	Understandability	Number of sections/subsections
	Volatility	Number of changes per requirement
		Time (by activity) when change is requested
	Traceability	Number of requirements not traceable to design/code
	Model clarity	Number of UML models
		Number of descriptive pages per model
		Number of UML errors
Design quality	Architectural integrity	Existence of architectural model
	Component completeness	Number of components that trace to architectural model
		Complexity of procedural design
	Interface complexity	Average number of pick to get to a typical function or content
		Layout appropriateness
	Patterns	Number of patterns used
Code quality	Complexity	Cyclomatic complexity
	Maintainability	Design factors (Chapter 8)
	Understandability	Percent internal comments
		Variable naming conventions
	Reusability	Percent reused components
	Documentation	Readability index
QC effectiveness	Resource allocation	Staff hour percentage per activity
	Completion rate	Actual vs. budgeted completion time
	Review effectiveness	See review metrics (Chapter 14)
	Testing effectiveness	Number of errors found and criticality
		Effort required to correct an error
		Origin of error

SQA Goals (see Figure 16.1)

- Process metrics
  - Software process quality metrics
  - Software process timetable metrics
  - Error removal effectiveness metrics
  - Software process productivity metrics
- Software process quality metrics
  - Error density metrics
  - Error severity metrics

SQA Goals (see Figure 16.1)

- Error density metrics
  - number of code errors (NCE)
  - weighted number of code errors (WCE)

Error severity class *a*	Calculation of NCE (number of errors) *b*	Calculation of WCE	
		Relative weight *c*	Weighted errors $D = b \times c$
Low severity	42	1	42
Medium severity	17	3	51
High severity	11	9	99
Total	70	—	192
NCE	70	—	—
WCE	—	—	192

SQA Goals (see Figure 16.1)

- Error density metrics

Code	Name	Calculation formula
CED	Code Error Density	$CED = \frac{NCE}{KLOC}$
DED	Development Error Density	$DED = \frac{NDE}{KLOC}$
WCED	Weighted Code Error Density	$WCED = \frac{WCE}{KLOC}$
WDED	Weighted Development Error Density	$WDED = \frac{WDE}{KLOC}$
WCEF	Weighted Code Errors per Function point	$WCEF = \frac{WCE}{NFP}$
WDEF	Weighted Development Errors per Function point	$WDEF = \frac{WDE}{NFP}$

SQA Goals (see Figure 16.1)

- Error density metrics

- NCE = number of code errors detected in the software code by code inspections and testing. Data for this measure are culled from code inspection and testing reports.
- KLOC = thousands of lines of code.
- NDE = total number of development (design and code) errors detected in the software development process. Data for this measure are found in the various design and code reviews and testing reports conducted.
- WCE = weighted code errors detected. The sources of data for this metric are the same as those for NCE.
- WDE = total weighted development (design and code) errors detected in development of the software. The sources of data for this metric are the same as those for NDE.
- NFP = number of function points required for development of the software. Sources for the number of function points are professional surveys of the relevant software.

SQA Goals (see Figure 16.1)

- Error density metrics

Measures and metrics	Calculation of CED (Code Error Density)	Calculation of WCED (Weighted Code Error Density)
NCE	70	—
WCE	—	192
KLOC	40	40
CED (NCE/KLOC)	1.75	—
WCED (WCE/KLOC)	—	4.8

SQA Goals (see Figure 16.1)

- *Error seveity metrics*

Code	Name	Calculation formula
ASCE	Average Severity of Code Errors	$ASCE = \frac{WCE}{NCE}$
ASDE	Average Severity of Development Errors	$ASDE = \frac{WDE}{NDE}$

SQA Goals (see Figure 16.1)

- Software process timetable metrics

Code	Name	Calculation formula
TTO	Time Table Observance	$TTO = \frac{MSOT}{MS}$
ADMC	Average Delay of Milestone Completion	$ADMC = \frac{TCDAM}{MS}$

Key:

- MSOT = milestones completed on time.
- MS = total number of milestones.
- TCDAM = total Completion Delays (days, weeks, etc.) for All Milestones. To calculate this measure, delays reported for all relevant milestones are summed up. Milestones completed on time or before schedule are considered "O" delays. Some professionals refer to completion of milestones before schedule as "minus" delays. These are considered to balance the effect of accounted-for delays (we might call the latter "plus" delays). In these cases, the value of the ADMC may be lower than the value obtained according to the metric originally suggested.

SQA Goals (see Figure 16.1)

- Error removal effectiveness metrics

Code	Name	Calculation formula
DERE	Development Errors Removal Effectiveness	$DERE = \frac{NDE}{NDE + NYF}$
DWERE	Development Weighted Errors Removal Effectiveness	$DWERE = \frac{WDE}{WDE + WYF}$

Key:
- NYF = number of software failures detected during a year of maintenance service.
- WYF = weighted number of software failures detected during a year of maintenance service.

SQA Goals (see Figure 16.1)

- Software process productivity metrics

Code	Name	Calculation formula
DevP	Development Productivity	$DevP = \frac{DevH}{KLOC}$
FDevP	Function point Development Productivity	$FDevP = \frac{DevH}{NFP}$
CRe	Code Reuse	$CRe = \frac{ReKLOC}{KLOC}$
DocRe	Documentation Reuse	$DocRe = \frac{ReDoc}{NDoc}$

Key:
- DevH = total working hours invested in the development of the software system.
- ReKLOC = number of thousands of reused lines of code.
- ReDoc = number of reused pages of documentation.
- NDoc = number of pages of documentation.

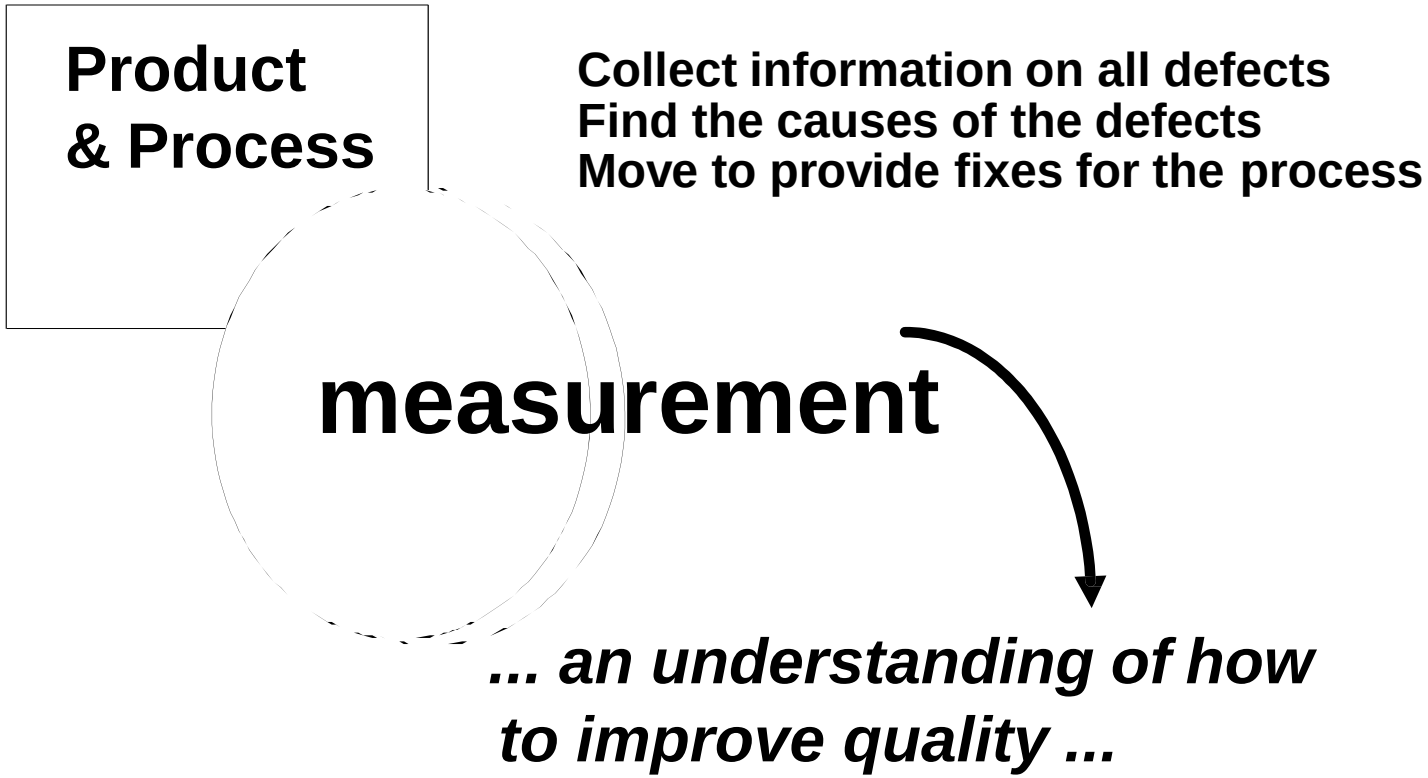
SQA Goals (see Figure 16.1)

- Software process productivity metrics

Code	Name	Calculation formula
DevP	Development Productivity	$DevP = \frac{DevH}{KLOC}$
FDevP	Function point Development Productivity	$FDevP = \frac{DevH}{NFP}$
CRe	Code Reuse	$CRe = \frac{ReKLOC}{KLOC}$
DocRe	Documentation Reuse	$DocRe = \frac{ReDoc}{NDoc}$

Key:
- DevH = total working hours invested in the development of the software system.
- ReKLOC = number of thousands of reused lines of code.
- ReDoc = number of reused pages of documentation.
- NDoc = number of pages of documentation.

Statistical SQA



Statistical SQA

- Information about software errors and defects is collected and categorized.
- An attempt is made to trace each error and defect to its underlying cause (e.g., non-conformance to specifications, design error, violation of standards, poor communication with the customer).
- Using the Pareto principle (80 percent of the defects can be traced to 20 percent of all possible causes), isolate the 20 percent (the *vital few*).
- Once the vital few causes have been identified, move to correct the problems that have caused the errors and defects.

Six-Sigma for Software Engineering

- The term “six sigma” is derived from six standard deviations—3.4 instances (defects) per million occurrences—implying an extremely high quality standard.
- The Six Sigma methodology defines three core steps:
  - *Define* customer requirements and deliverables and project goals via well-defined methods of customer communication
  - *Measure* the existing process and its output to determine current quality performance (collect defect metrics)
  - *Analyze* defect metrics and determine the vital few causes.
  - *Improve* the process by eliminating the root causes of defects.
  - *Control* the process to ensure that future work does not reintroduce the causes of defects.

Software Reliability

- A simple measure of reliability is *mean-time-between-failure* (MTBF), where

$$MTBF = MTTF + MTTR$$

- The acronyms MTTF and MTTR are *mean-time-to-failure* and *mean-time-to-repair*, respectively.
- *Software availability* is the probability that a program is operating according to requirements at a given point in time and is defined as

$$Availability = [MTTF/(MTTF + MTTR)] \times 100\%$$

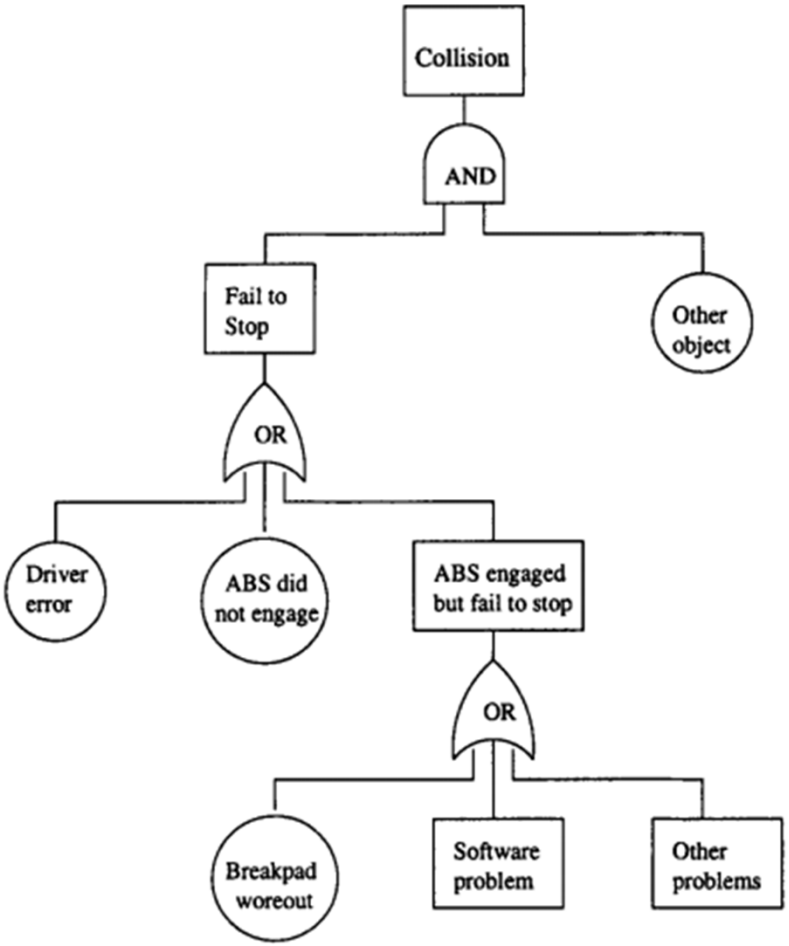
Software Reliability

- An alternative measure of reliability is *failures-in-time* (FIT)—a statistical measure of how many failures a component will have over 1 billion hours of operation. Therefore, 1 FIT is equivalent to one failure in every billion hours of operation.

Software Safety

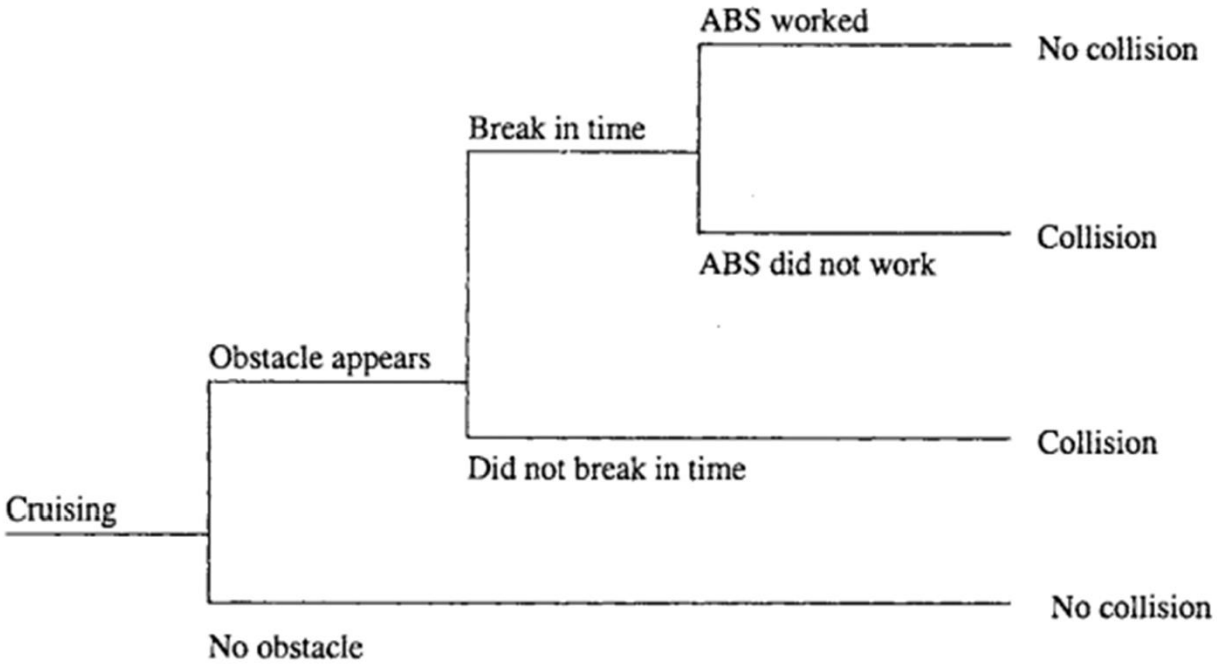
- *Software safety* is a software quality assurance activity that focuses on the identification and assessment of potential hazards that may affect software negatively and cause an entire system to fail.
- If hazards can be identified early in the software process, software design features can be specified that will either eliminate or control potential hazards.

Software Safety



These slides are designec
(McGraw-Hill 2009). Slide

Software Safety



ISO 9000 Standard

Motivation for ISO 9000 Implementation
Enhances the credibility of the company
Marketing benefit of ISO 9000 certificate
Indicates that customer satisfaction and quality are a core value of the company
Indicates that the company is dedicated to continuous improvement
Indicates that the company plans for quality in product development
Higher quality software produced
Indicates that a fire prevention culture rather than a fire fighting culture is in place
The emphasis is on learning lessons from problems
A more capable and effective organization
More loyal customers
Protects the company from litigation. The ISO 9000 requirements on records provide evidence that all reasonable steps were taken
ISO 9000 offers a framework or model to improve. The standard may be used by the organization to improve
Less re-work of defective products
Higher morale in the organization

ISO 9000 Standard

Clause	Description
Quality Management System	This clause refers to the documentation and implementation of the quality management system.
Resource management	This is concerned with the provision of the resources needed to implement the quality management system.
Product or service realisation	The provision of processes to implement the product or service.
Management Responsibility	The responsibility of management in the implementation of the quality management system.
Measurement, analysis and improvement	The establishment of a measurement program to measure the quality management system performance and to identify improvements.

ISO 9000 Standard

The structure of ISO 9001:2008