1 Fundamentals of Software Quality

Key Topics

Shewhart
Deming
Juran
Crosby
Watts Humphrey
Metrics
Problem-solving
Cost of quality
Process improvement
Customer satisfaction

1.1 Introduction

The mission of a software company is to develop high-quality innovative products and services at a competitive price to its customers and to do so ahead of its competitors. This requires a clear vision of the business, a culture of innovation, an emphasis on quality, detailed knowledge of the business domain, and a sound product development strategy.

It requires a focus on software quality and customer satisfaction, and quality must be built into the software product so that customers remain loyal to the company. Customers have very high expectations on quality and expect high-quality software products to be consistently delivered on time and on budget.

G. O'Regan, *Concise Guide to Software Testing*,
Undergraduate Topics in Computer Science,
https://doi.org/10.1007/978-3-030-28494-7_1

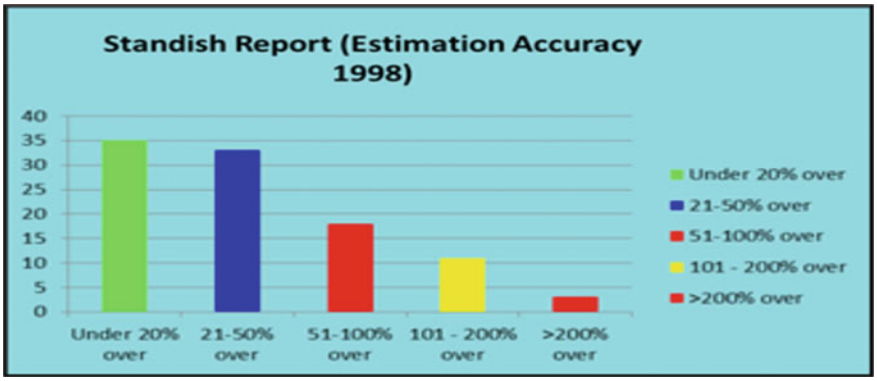


Fig. 1.1 Standish research—project cost estimation accuracy in 1998

The focus on quality requires effective software processes to be in place so that quality software may be consistently produced.

Software testing plays a key role both in building quality into the software and in verifying that the desired quality has been achieved. Quality improvement is essential, and a focus on industrial best practice and emerging technologies assists in performance improvement.

The history of quality and some of the key people who have contributed to the quality movement are discussed later in the chapter. This includes well-known quality gurus such as Shewhart, Deming, Juran, and Crosby. These figures played an important role in promoting quality and transforming struggling manufacturing companies. Watts Humphrey is considered the father of software quality, and his important contributions to software process improvement are discussed.

The Standish Group Research (1999) (Fig. 1.1) on project cost overruns in the US during 1998 indicate that 33% of projects were between 21 and 50% over estimate, 18% were between 51 and 100% over estimate, and 11% of projects were between 101 and 200% overestimate.[1]

Projects sometimes fail, and there are many examples of projects being abandoned prior to completion. For example, the Taurus project at the London stock exchange is a well-known disaster. The project was eventually abandoned, and at that stage, it was 11 years late and had cost the city of London hundreds of millions of pounds (Manley 1995).

It is essential that requirements are properly managed as uncontrolled changes to requirements may have a negative effect on the project. It may be necessary to accept a late change to the requirements, but there are corresponding risks to the project schedule and quality. However, a good requirements process will ensure that changes to the requirements are minimized and controlled, and the

[1]The study was from the mid/late 1990 and recent reports from the Standish Group show good improvement trends.

requirements process will often include prototyping or joint user reviews to ensure that the requirements are actually those desired by the customer.

The implementation of the requirements involves design, development, and testing activities. It may also involve the production of user manuals and training materials as well as the technical documentation. Quality must be built into the software, and the goal of the testing activities is to verify the correctness of the software. The project manager is responsible for delivering the project on time and for recovering the schedule when it falls behind.

Engineers have constructed bridges for several millennia, and bridge building is considered a mature engineering activity. However, occasionally civil engineering projects fall behind schedule or suffer design flaws. For example, the infamous Tacoma Narrows Bridge (or Galloping Gertie as it was known) collapsed in 1940 due to a design flaw.

The Tacoma Narrows Bridge was known for its tendency to sway in windstorms. The shape of the bridge was like that of an aircraft wing, and under windy conditions, it would generate sufficient lift to become unstable. A large windstorm in November 1940 caused catastrophic failure. The significance of the Tacoma Bridge is its collapse and the subsequent investigation by engineers. They realized that aerodynamical forces in suspension bridges were not sufficiently understood in the design of the bridge and that new research was needed. It was recommended that wind tunnel tests be used to aid in the design of the replacement bridge.

Software engineering is a less mature field than civil engineering, and it is only in more recent times that investigations and recommendations from software projects have become part of the software development process. The study of software engineering has led to new theories and understanding of software development.

1.2 History of Software Failures

There are many examples of software failures in the literature. These include the year 2000 (or Y2K) problem which was a design flaw in the representation of the date with two digits; the Intel Pentium microprocessor bug which referred to a floating-point problem on an Intel microprocessor back in 1994; the Ariane 5 launcher disaster was due to an operand error that resulted from the conversion of a 64-bit floating-point number to a 16-bit signed integer number. Software failures may cause major problems and adversely affect the customer's business. They may lead to credibility issues and damage to the customer relationship.

The Y2K bug is historical and part of computer science folklore. The event on 1 January 2000 had minimal impact on the world economy. However, organizations spent large sums of money in identifying all code with a year 2000 impact; changing the representation of the date from 2 digits to 4 digits; and verifying the correctness of the changes made. The worldwide cost of this was in billions of dollars.

The Intel response to a famous microprocessor bug back in 1994 inflicted temporary damage on the reputation of the company. Intel was slow to acknowledge the floating-point problem and in providing adequate information. This led to damage in its reputation and hundreds of millions of dollars to replace the flawed microprocessors.

The Ariane 5 led to major embarrassment and damage to the credibility of the European Space Agency (ESA). The maiden flight of the Ariane 5 launcher ended in failure on 4 June 1996, after a flight time of 40 s. The first 37 s of flight proceeded as normal. However, the launcher then veered off its flight path, broke up, and exploded. An independent inquiry board investigated the cause of the failure, and the report and recommendations to prevent a future failure are described in Lions (1996).

The inquiry noted that the failure of the inertial reference system was followed immediately by a failure of the backup inertial reference system. The problem was traced to a software failure due to an operand error resulting from the conversion of a 64-bit floating-point number to a 16-bit signed integer value number. The floating-point number was too large to be represented in the 16-bit number, and this resulted in the operand error.

The inertial reference system and the backup reference system reported failure due to the software exception. The operand error occurred owing to an exceptionally high value related to the horizontal velocity, and this was due to the fact that the early part of the trajectory of the Ariane 5 differed from the earlier Ariane 4, and required a higher horizontal velocity. The inquiry board made a series of recommendations to prevent a reoccurrence of similar problems.

These failures indicate that software quality needs to be a key driving force in any organization. The effect of software failure may result in huge costs to correct the software (e.g. Y2K), negative perception of a company and large replacement costs (e.g. Intel microprocessor problem), or the loss of a valuable communications satellite and all the costs associated with this (e.g. Ariane 5).

1.3 Background to Software Quality

Customers today have very high-quality and reliability expectations and expect companies to adhere to very high standards. There are many quality software products in the marketplace; however, the task of consistently producing high-quality software products is non-trivial. Even the most respected organizations occasionally deliver software that contains defects, or ship products late due to quality problems. Defects may cause minor irritation to a customer, loss of credibility, or lead to injury or loss of life.

The late delivery of a product leads to extra costs, and it may adversely affect the customer's revenue, profitability, and business planning. Consequently, it is essential to have a robust process to consistently develop high-quality software on

Table 1.1 ISO 9126 quality characteristics

Characteristic	Description
Functionality	This indicates the extent to which the required functionality is available in the software
Reliability	This indicates the extent to which the software is reliable
Usability	This indicates the extent to which the users of the software judge it to be easy to use
Efficiency	This characteristic indicates the efficiency of the software
Maintainability	This indicates the extent to which the software product is easy to modify and maintain
Portability	This indicates the ease of transferring the software to a different environment

time and within budget. The influential papers by Fred Brooks in Brooks (1975, 1986) suggest that there is no silver bullet to do this, and that instead, the focus needs to be on incremental improvement to processes and tools.

1.3.1 What Is Software Quality?

There are various definitions of quality such as the definition proposed by Philip Crosby as "*conformance to the requirements*". This definition does not take the intrinsic difference in quality of products into account in judging the quality of the product. For example, this definition might suggest that a *Mercedes* car is of the same quality as a *Lada* car.[2] Further, the definition does not consider whether the requirements are actually appropriate for the product.

Juran defines quality as "*fitness for use*", and this is a better definition, although it does not provide a mechanism to judge better quality when two products are equally fit to be used. The ISO 9126 standard for information technology (ISO/IEC 1991) is a framework for the evaluation of software product quality. It defines six product quality characteristics (Table 1.1), which indicate the extent to which a software product may be judged to be of a high quality by the customers.

1.3.2 Early Quality Management

In the Middle Ages, a craftsman was responsible for the complete development of a product from its conception to delivery to the customer. This led to a strong sense of pride and ownership of the quality of the product, and apprentices joined craftsmen to learn the skills of the trade.

The Industrial Revolution led to a change to this traditional paradigm, and labour became highly organized with workers responsible for a particular part of the

[2]Most rational people would judge the Mercedes to be of superior quality.

manufacturing process. The sense of ownership and the pride of workmanship in the product were diluted, as workers were now responsible only for their portion of the product, and not the quality of the product as a whole.

This led to a requirement for more stringent management practices, including planning, organizing, implementation, and control. It inevitably led to a hierarchy of labour with various functions identified and a reporting structure for the various functions. Supervisor controls were needed to ensure that quality and productivity issues were addressed.

1.3.3 Total Quality Management

Total quality management (TQM) is a modern approach to quality management, and this management philosophy involves customer focus, process improvement, developing a culture of quality within the organization and developing a measurement and analysis program. It emphasizes that customers have rights and quality expectations, which should be satisfied, and that everyone in the organization is both a customer and has customers.

It is a *holistic approach* and requires that all functions, in the organization, follow high standards. Quality needs to be built into the product by ensuring that quality is addressed at every step in the process.

It requires total commitment from the top management, and that all staff be trained in quality management and participate in quality improvement. It requires that a commitment to quality be instilled in all staff, and that the focus within the organization changes from *firefighting* to *fire prevention*. Problem-solving is used to identify the root causes of problems, and corrective action is taken to prevent their re-occurrence.

1.3.4 Software Quality Control

Software quality control is concerned with activities to ensure that the end product satisfies the functional and non-functional requirements and is fit for purpose. It includes inspections and testing to verify that the deliverables produced satisfy their requirements. Inspections typically consist of a formal review of a deliverable by independent experts, and the objective is to identify defects within the work product and to provide confidence in its correctness.

Inspections in a manufacturing environment are quite different in that they take place at the end of the production cycle and do not offer a mechanism to build quality into the product. Instead, the defective products are removed from the batch and reworked. There is a growing trend towards quality sampling at the early phases of a manufacturing process to minimize reworking of defective products.

Software testing consists of "*white box*" or "*black box*" testing techniques, and the testing activities include *unit*, *system, performance*, and *acceptance testing*. The testing is quite methodical and includes a comprehensive set of manual or automated test cases. The *verification* and *validation* activities involve the execution of the defined tests and the correction of any failed or blocked tests.

The cost of correction of a defect is related to the phase in which it is detected in the lifecycle. Errors detected in phase are the least expensive to correct, and defects detected out of phase become increasingly expensive to correct. The most expensive defect is that of a requirements' defect identified by the customer, as its correction may involve changes to the requirements, design, and code. Testing will be required as well as a fix release for the customer. There is further overhead in project management, configuration management, and in communication with the customer.

It is, therefore, highly desirable to capture defects as early as possible in the software lifecycle to minimize the effort required to correct. Modern software engineering places emphasis on defect prevention and in learning lessons from the defects. This approach is adopted from manufacturing environments and consists of formal causal analysis meetings to brainstorm and identify root causes of problems and to define the corrective actions necessary to prevent reoccurrence. The actions are then implemented and tracked to completion.

1.4 History of Quality

This section considers the ideas of several pioneers who have influenced the quality field. These include Walter Shewhart, W. Edwards Deming, Joseph Juran, and Philip Crosby. We also discuss the influence of Watts Humphrey who is considered the father of software quality.

1.4.1 Shewhart

Walter Shewhart was Statistician at AT&T Bell Laboratories (or Western Electric Co. as it was known in the 1920s). He is regarded as Founder of statistical process control (SPC), which remains important today in monitoring and controlling a process (Fig. 1.2). Shewhart developed a control chart, which is used to control the process, with upper and lower limits for process performance specified. The process is under control if it is performing within these limits.

Shewhart's ideas were applied to the Capability Maturity Model (CMM) in the late 1980s as a way to control key software processes. Statistical process control (SPC) plays an important role in process improvement and in ensuring that process performance is acceptable. It is used to minimize variability in process performance, as variability in the process affects product quality. SPC involves the analysis of control charts so that the cause of variability can be identified and eliminated. Deming and Juran worked with Shewhart at Bell Labs in the 1920s.

The Shewhart model is a systematic approach to problem-solving and process control. It consists of four steps that are used for continuous process improvement,

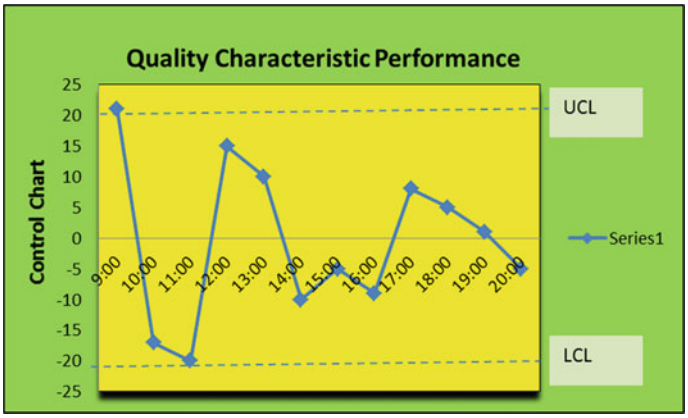
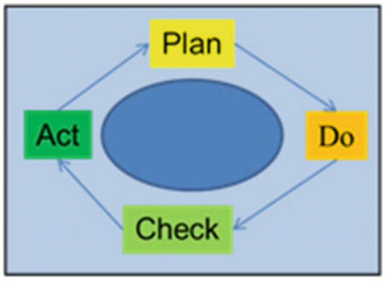


Fig. 1.2 Shewhart's control chart

Fig. 1.3 Shewhart's PDCA cycle



which are *plan*, *do*, *check*, *act* (Fig. 1.3). It is known as the *"PDCA model"* or Shewhart's model and is described in Table 1.2.

Shewhart argued that quality and productivity improve as process variability is reduced. His influential book, *The Economic control of quality of manufactured product* (Shewhart 1931), outlines the methods of statistical process control to reduce process variability. It predicted that productivity would improve as process variability was reduced, and this was verified by Japanese engineers in the 1950s.

This led to productivity improvements and increased market share for Japanese companies. Today, companies around the world recognize the importance of placing quality at the heart of the organization.

1.4.2 Deming

W. Edwards Deming (Fig. 1.4) was a major figure in the quality movement. He was influenced by Shewhart's work on statistical process control, and Deming's approach was adopted in post-Second World War Japan. He played an important role in transforming Japan industry.

Table 1.2 Shewhart cycle

Step	Description
Plan	This step identifies an improvement opportunity and outlines the problem or process that will be addressed – Select the problem to be solved – Describe current process – Identify the possible causes of the problem – Find the root cause – Develop an action plan to correct the root cause
Do	This step involves carrying out the improvement actions, and it may involve a pilot of the proposed changes to the process
Check	This step involves checking the results obtained to determine their effectiveness
Act	This step includes the analysis of the results to adjust process performance to achieve the desired results

Fig. 1.4 W.E. Deming

Deming argued that it is not sufficient for everyone in the organization to be doing one's best: instead, what is required is that there be a consistent purpose and direction in the organization. That is, it is first necessary that people know what to do, and there must be a *constancy of purpose* from all individuals to ensure success.

He argued that there is a very strong case for improving quality, as costs will decrease due to less rework, and productivity will increase as less time is spent in reworking defective products. This will enable the company to increase its market share, with better quality and lower prices, and to stay in business. Conversely, companies that fail to address quality issues will lose market share and go out of business. Deming was highly critical of the then American approach to quality and the lack of vision of American management in quality management.

Deming's influential book *Out of the Crisis* (Deming 1986) proposed 14 principles to transform the western style of management of an organization to a quality- and customer-focused organization. These include:

- Constancy of purpose
- Quality built into the product
- Continuous improvement culture.

Deming's ideas are described in more detail in Table 1.3.

Deming argued that there are several diseases that afflict companies in the western world that prevent them for achieving high-quality results. The *"five deadly diseases"* noted by Deming include (Table 1.4).

Comment (Deming)
Deming's programme has been quite influential and has many sound points. His views on slogans in the workplace are in direct opposition to the use of slogans like Crosby's "zero defects". The key point for Deming is that a slogan has no value unless there is a clear method to attain the particular goal described by the slogan.

1.4.3 Juran

Joseph Juran (Fig. 1.5) was a major figure in the quality movement, and he argued for a top-down approach to quality. He defined quality as *"fitness for use"* and argued that quality issues are the direct responsibility of management. Management must ensure that quality is planned, controlled, and improved.

The trilogy of *quality planning*, *control*, and *improvement* is known as the *"Juran Trilogy"* and is usually described by a diagram with time on the horizontal axis and the cost of poor quality on the vertical axis (Fig. 1.6).

Quality planning consists of setting quality goals, developing plans, and identifying the resources needed to achieve the goals. Quality control consists of evaluating performance, setting new goals, and taking appropriate action. Quality improvement consists of improving delivery, eliminating wastage, and improving customer satisfaction. Juran's 10-step programme for quality planning is defined in Juran (1951) and is summarized in Table 1.5.

Juran defined an approach to achieve a new quality performance level that is termed *"Breakthrough and Control"*. It is described pictorially by a control chart showing the old performance level with occasional spikes or random events; what is needed is a breakthrough to a new and more consistent quality performance, i.e. a new performance level with the performance achieved at that level.

The example in Fig. 1.7 presents a breakthrough in developing a more accurate estimation process. Initially, the variation in estimation accuracy is quite large, but as an improved estimation process is put in place, the control limits are narrowed and more consistent estimation accuracy is achieved.

The breakthrough is achieved by a sustained and coordinated effort, and the old performance standard becomes obsolete. The difference between the old and the new performance level is known as the *"chronic disease"* which must be diagnosed and cured. His approach to breakthrough and control is described in Table 1.6.

Table 1.3 Deming 14-step programme

Step	Description
Constancy of purpose	Companies face short-term and long-term problems. The problems of tomorrow require long-term planning on new products, training, and innovation. This requires R&D and continuous improvement of existing products and services
Adopt new philosophy	Deming outlined the *five deadly diseases* that afflicted US companies. These included lack of purpose and an excessive interest in short-term profits
Build quality in	Deming argued that performing mass inspections is equivalent to planning for defects, as they are too late to improve quality. Consequently, it is necessary to improve the production process to *build the quality into* the product
Price and quality	Deming argued against awarding business on price alone, as the price is meaningless unless there is an objective measure of the quality of the product being purchased
Continuous improvement	There must be *continuous improvement in all areas*, including understanding customer requirements, design, manufacturing, and test methods
Institute training	The organization must be a learning organization with a training programme to educate management and staff about the company, customer needs, and pride of workmanship in the products. Supervisors and managers need training on the 14-point program
Institute leadership	Deming argues that *management is about leadership and not supervision*. Management should work to remove barriers, know the work domain, and seek innovative solutions to resolve quality and other relevant issues
Eliminate fear	The presence of fear is a barrier to an open discussion of problems and the identification of solutions or changes to prevent problems from arising
Eliminate barriers	The objective here is to break down barriers between different departments and groups. It is not enough for each group to optimize its own area: instead, what is required is for the organization to be working as one team
Eliminate slogans	Deming argued that slogans do not help anyone to do a better job. Slogans may potentially alienate staff or encourage cynicism. Deming criticized slogans such as "*Zero Defects*" or "*Do it right the first time*" as inappropriate, as how can it be made right first time if the production machine is defective. Most problems are due to the system rather than the person
Eliminate numerical quotas	Deming argued that quotas act as an impediment to improvement in quality, as quotas are normally based on what may be achieved by the average worker. People below the average cannot make the rate, and the result is dissatisfaction and turnover. Thus, there is a fundamental conflict between quotas and pride of workmanship
Pride of work	The intention here is to remove barriers that rob people of pride of workmanship (e.g. machines out of order)

(continued)

Table 1.3 (continued)

Step	Description
Self improvement	This involves encouraging education and self-improvement for everyone in the company
Take action	This requires that management agree on direction using the 14 principles, communicate the reasons for changes to the staff, and train the staff on the 14 principles

Table 1.4 Deming—five deadly diseases

Disease	Description
Lack of constancy of purpose	Management is too focused on short-term thinking rather than long-term improvements
Emphasis on short-term profit	A company should aim to become the world's most efficient provider of product/service. Profits will then follow
Evaluation of performance	Deming is against annual performance appraisal and rating
Mobility of management	Mobility of management frequently has a negative impact on quality
Excessive measurement	Excessive management by measurement

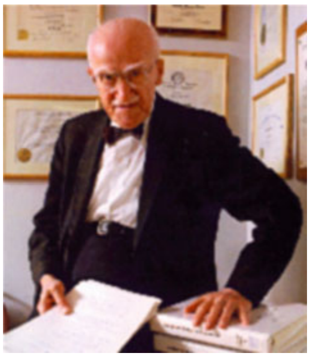


Fig. 1.5 Joseph Juran

1.4.4 Crosby

Philip Crosby was a key figure in the quality movement, and his quality improvement grid later influenced the design of the Capability Maturity Model (CMM), which was developed by the Software Engineering Institute. His influential book *Quality is Free* (Crosby 1979) outlines his philosophy of *doing things right the first time*, i.e. the *zero defects* (ZD) program. Quality is defined as *"conformance to the requirements"*, and he argues that people have been conditioned to believe that error is inevitable.

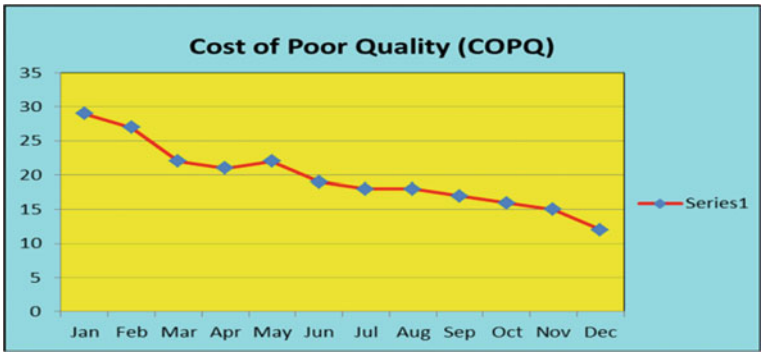


Fig. 1.6 Cost of poor quality—% of sales

Table 1.5 Juran's 10-step programme for quality planning

Step	Description
Identify customers	This includes the internal and external customers of an organization; e.g., the testing group is an internal customer, whereas the end-user of the software is an external customer
Determine customer needs	Customer needs are generally expressed in the language of the customer's organization. There is a need to elicit and determine the actual desired requirements from discussion and communication with the customer
Translate	This involves translating the customer needs into the language of the supplier
Units of measurement	This involves defining the measurement units to be used
Measurement programme	This involves setting up a measurement programme in the organization, and it includes internal and external measurements of quality and process performance
Develop product	This step determines the product features to meet the needs of the customer
Optimize product design	The intention is to optimize the design of the product to meet the needs of the customer and supplier
Develop process	This involves developing processes that can produce the products to satisfy the customer's needs
Optimize process capability	This involves optimizing the capability of the process to ensure that high-quality products are produced
Transfer	This involves transferring the process to normal product development operations

Crosby argued that people in their personal lives do not accept this: for example, it would not be acceptable for nurses to drop a certain percentage of newly born babies. He further argues that the term *"acceptable quality level"* (AQL) is a commitment to produce imperfect material. Crosby notes that defects are due to two main reasons: *lack of knowledge* or a *lack of attention of the individual*.

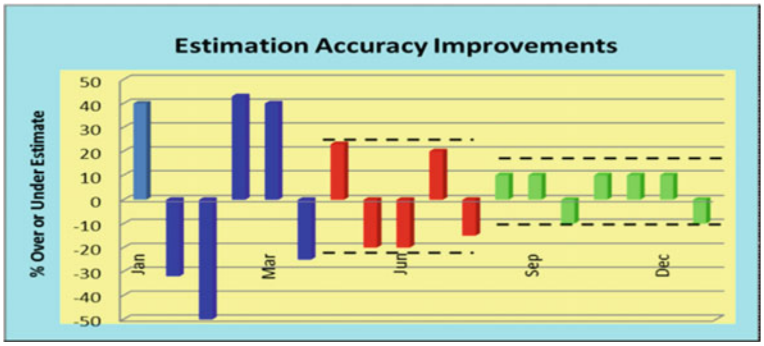


Fig. 1.7 Estimation accuracy—breakthrough and control

Table 1.6 Juran's breakthrough and control

Step	Description
Breakthrough in attitude	This involves developing a favourable attitude to quality improvement
Pareto	This involves identifying the key areas affecting quality
Organization	This involves analysing the problem and coordinating a solution
Control	This is concerned with achieving performance at the new level
Repeat	This leads to continuous improvement with new performance levels set, and new breakthroughs made to achieve higher performance levels

He argued that lack of knowledge can be measured and addressed by training, but that lack of attention is a mindset that requires a change of attitude by the individual. The net effect of a successful implementation of a zero defects programme is higher productivity due to less reworking of defective products. Thus, *quality*, in effect, *is free*.

Crosby's approach to achieving the desired quality level of zero defects was to put a quality improvement programme in place. He outlined a 14-step quality improvement programme (Table 1.7). It requires management commitment to be successful, and an organization-wide quality improvement team needs to be set up. A measurement programme is put in place to determine the status and cost of quality within the organization. The cost of quality is then shared with the staff, and corrective actions are identified and implemented. The zero defects programme is communicated to the staff, and one day every year is made a *zero defects day* and is used to emphasize the importance of zero defects to the organization.

Crosby's Quality Management Maturity Grid (Table 1.8) measures the maturity of the current quality system with respect to several quality management categories and highlights areas that require improvement. Six categories of quality management are considered: *management understanding and attitude towards quality, quality organization status, problem handling, the cost of quality, quality improvement actions, and summation of company quality posture.*

Table 1.7 Crosby's 14-step programme

Step	Description
Management commitment	Management commitment and participation are essential to the success of the quality improvement program. The profile of quality is raised within the organization
Quality improvement team	This involves the formation of an organization-wide cross-functional team consisting of representatives from each of the departments
Quality measurement	The objective is to determine the status of quality in each area of the company to identify areas where improvements are required
Cost of quality evaluation	The cost of quality indicates the financial cost of quality to the organization. It is initially high, but reduces as the quality improvement programme becomes effective
Quality awareness	This involves sharing the cost of poor quality with staff and motivating staff to identify corrective actions to deal with quality issues
Corrective action	This involves resolving any problems that have been identified and bringing any problems that cannot be resolved to the attention of management
Zero defects program	The key point is that zero defects is not a motivation program: instead, it means doing things right the first time, i.e. zero defects
Supervisor training	This requires that all supervisors and managers receive training on the 14-step quality improvement program
Zero defects day	This involves setting aside one day each year to high-light zero defects and its importance to the company
Goal setting	This phase involves getting people to think in terms of goals and how the goals may be achieved
Error cause removal	This involves removing any roadblocks or problems that prevent employees from performing error-free work
Recognition	This involves recognizing employees who make outstanding contributions to quality improvement
Quality councils	This involves bringing quality professionals together on a regular basis to share ideas on quality
Do it over again	The principle of continuous improvement is a key part of the programme, as improvement is continuous

Each category is rated on a 1-to-5 maturity scale which indicates the maturity of the particular category. Crosby's maturity grid was later adapted and applied to the CMM. The five maturity levels of Crosby's grid are:

Comment (Crosby)
Crosby's programme has been quite influential, and his maturity grid has been applied to the software CMM. The ZD part of the programme is difficult to apply to the complex world of software development, where the complexities of the systems to be developed are often the cause of defects rather than the mindset of software professionals (who are generally professional and dedicated to quality). Slogans may be dangerous and potentially unsuitable to some cultures, and a zero defects day may potentially have the effect of de-motivating staff.

Table 1.8 Crosby's maturity grid

Level	Name	Description
1.	Uncertainty	Management has no understanding of quality and is likely to blame quality problems on the quality department. Firefighting is prevalent, and problems are fought as they occur. Root causes of problems are not investigated, and there are few organized quality improvement activities
2.	Awakening	Management is beginning to recognize that quality management may be of value, but is unwilling to devote time and money to it. Instead, the emphasis is on appraisal rather than prevention. Teams are set up to address major problems, but long-term solutions are rarely sought
3.	Enlightenment	Management is learning more about quality and is becoming more supportive of quality improvement. The quality department reports to senior management, and implementation of the 14-step quality improvement programme is underway. There is a culture of openness where problems are faced openly and resolved in an orderly way
4.	Wisdom	Management is fully participating in the program and fully understands the importance of quality management. All functions within the organization are open to suggestions for improvement, and problems are identified earlier. Defect prevention is now part of the culture
5.	Certainty	The whole organization is involved in continuous improvement

1.4.5 Watts Humphrey

Watts Humphrey was an American software engineer and vice-president of technical development at IBM. He made important contributions to the software engineering field and is considered the *father of software quality*. He dedicated much of his career to addressing the problems of software development including schedule delays, cost overruns, software quality, and productivity (Fig. 1.8).

He was born in Michigan in 1927 and served in the US Navy and completed a bachelor's degree in physics at the University of Chicago in 1949. He obtained a

Fig. 1.8 Watts Humphrey. Courtesy of Watts Humphrey

master's degree in physics from the Illinois Institute of Technology (IIT) and an MBA from the University of Chicago.

He took a position with Sylvania in Boston in the early 1950s, and he became Manager of the circuit design group in the company. He recognized the importance of planning and management early in his career, and he joined IBM in 1959 initially as Hardware Architect, but most of his IBM career was in management. He was eventually to become Vice-President of technical development, where he oversaw 4,000 engineers in 15 development centres in over 7 countries. Others at IBM influenced him including Fred Brooks who was Project Manager of the IBM 360 project; Michael Fagan who developed the Fagan inspection methodology; and Harlan Mills who developed the Cleanroom methodology. Humphrey ran the software quality and process group at IBM towards the end of his IBM career and became very interested in software quality.

He retired from IBM in 1986 and joined the newly formed SEI at Carnegie Mellon University. He made a commitment to change the software engineering world by developing sound management principles for the software industry. The SEI has largely fulfilled this commitment, and it has played an important role in enhancing the capability of software organizations throughout the world.

The SEI had a contract from the Department of Defence (DOD) to provide guidance to the military in the selection of capable software subcontractors. This evolved into the book "Managing the Software Process" (Humphry 1989) which describes technical and managerial topics essential for good software engineering. The book was influenced by the ideas of Deming and Juran in statistical process control.

Humphrey established the software process programme at the SEI, and this led to the development of the software Capability Maturity Model (CMM) and its successors. Humphrey asked questions such as:

– How good is the current software process?
– What must I do to improve it?
– Where do I start?

The CMM is a framework to help an organization to understand its current process maturity and to prioritize improvements. The SEI introduced software process assessment and software capability evaluation methods, and these include CBA/IPI and CBA/SCE. The CMM and the associated assessment methods were widely adopted by organizations around the world, and their successors are the CMMI Model and the SCAMPI appraisal methodology.

Humphrey focused his later efforts to developing the Personal Software Process (PSP) and the Team Software Process (TSP). These are approaches that teach engineers the skills they need to make and track plans and to produce high-quality software with zero defects. The PSP helps the individual engineer to collect relevant data for statistical process control, whereas the TSP focuses on teams, and the goal is to assist teams to understand and improve their current productivity and quality of their work.

He received many awards for his contributions to the computing field. He was named the first SEI fellow in 1995 in recognition of his outstanding contribution to the software quality field. He received the 2003 National Medal in Technology and Innovation from President George Bush, and he was named an ACM fellow in 2009 for his outstanding contributions to computing and information technology. He was the author of twelve books in the software engineering field, and he died in 2010.

1.4.6 Miscellaneous Quality Gurus

There are several other pioneers in the quality field including *Shingo* who developed his own version of zero defects termed *"Poka-yoke"* (or *defects* = 0). This involves identifying potential error sources in the process and monitoring these for errors. Causal analysis is performed on any errors found, and the root causes are eliminated. This approach leads to the elimination of all errors likely to occur, and thus only exceptional errors should occur. These exceptional errors and their causes are then eliminated. The failure mode and effects analysis (FMEA) methodology is a variant of this. Potential failures to the system or subsystem are identified and analysed, and the causes and effects and probability of failure documented.

Genichi Taguchi's definition of quality is quite different. Quality is defined as *"the loss a product causes to society after being shipped, other than losses caused by its intrinsic function"*. Taguchi defines a *loss function* as a measure of the cost of quality; $L(x) = c(x - T)^2 + k$. Taguchi also developed a method for determining the optimum value of process variables which will minimize the variation in a process while keeping a process mean on target.

Kaoru Ishikawa did work on *quality control circles* (QCCs). A quality control circle is a small group of employees who do similar work and meet regularly to identify and analyse work-related problems. This involves brainstorming, recommending, and implementing solutions. The problem-solving tools employed include *Pareto analysis*, *fishbone diagrams*, *histograms*, *scatter diagrams*, and *control charts*. A facilitator will train the quality circle team leaders, and the activities in a quality circle include:

- Select problem
- State and restate problem
- Collect facts
- Brainstorm
- Build on each other's ideas
- Choose course of action
- Presentation.

Armand Feigenbaum did work in *total quality control* which concerns quality assurance applied to all functions in the organization. It is distinct from total quality management: total quality control is concerned with controlling quality throughout, whereas TQM embodies a philosophy of quality management and improvement involving all staff and functions throughout the organization.

1.5 Modern Software Quality Management

The development of high-quality software requires a good software development process to be in place, and this includes best practices in software engineering for:

- Project management
- Estimation
- Risk management
- Requirements' development and management
- Design and development
- Software development lifecycles
- Quality assurance/management
- Software inspections
- Software testing
- Supplier selection and management
- Configuration management
- Customer satisfaction process
- Continuous improvement.

The cost of correction of a defect increases the later that it is detected in the lifecycle. Consequently, it is desirable to detect an error as early as possible and preferably within the phase in which it was created. Software inspections play a key role in detecting defects in-phase, and they are discussed in the next section.

1.5.1 Software Inspections

The Fagan inspection process was developed by Michael Fagan of IBM (Fagan 1976), and it aims to identify and remove errors in work products. The process mandates that requirement documents, design documents, source code, and test plans all be formally inspected by experts independent of the author of the deliverable.

There are various *roles* defined in the process including the *moderator* who chairs the inspection. The moderator ensures that all of the inspectors are trained and receive the appropriate materials for the inspection. He/she ensures that sufficient preparation is done, and that the speed of the inspection does not exceed the recommended guidelines. The *reader* reads or paraphrases the particular deliverable; the *author* is the creator of the deliverable and has a special interest in ensuring that it is correct. The *tester* role is concerned with the test viewpoint.

The inspection process will consider whether the design is correct with respect to the requirements, and whether the source code is correct with respect to the design. The errors identified are classified into various types, and the data is generally recorded to enable analysis to be performed on the most common types of errors to yield actions to minimize the re-occurrence of the most common defect types. Software inspections are described in more detail in Chap. 4.

1.5.2 Software Testing

Software testing plays a key role in verifying that the software is fit for purpose, and two key types of software testing are *black box* and *white box* testing. White box testing involves checking that every path in a module has been tested and involves defining and executing test cases to ensure code and branch coverage. The goal of black box testing is to verify the functionality of a module or feature or the complete system itself. Testing is both a constructive activity in that it is verifying the correctness of functionality, and it may be a destructive activity in that the objective is to find defects in the implemented software. Testing verifies that the requirements are correctly implemented, and it yields the presence or absence of defects.

There are various types of testing including unit, system, performance, and usability testing. The effectiveness of the testing is influenced by the maturity of the test process employed. Testing is described in detail in the remainder of this book.

1.5.3 Software Quality Assurance

The software quality assurance department provides visibility into the quality of the work products being built and the processes being used to create them. Its activities include audits of the various groups involved in software development.

The quality group promotes quality in the organization and is independent of the development group. It provides an independent assessment of the quality of the product being built, and this viewpoint is independent of the project manager and development viewpoint. The quality assurance group acts as the voice of the customer and aims to ensure that quality is considered at each step in the process.

The quality group will perform audits of various projects, groups, and departments and will determine the extent to which the process is followed and report any weaknesses in the processes and non-compliances identified. Any non-compliance issues that are not addressed may be escalated to the next level of management for resolution. Its key responsibilities are:

- Promotes quality in organization
- Conducts audits to verify compliance
- Reports audit results to management

- Provides visibility to management on processes followed
- Facilitates software process improvement
- Release sign-offs.

The quality audit provides visibility into the work products and processes used to develop the work products. The audit consists of an interview with the project team, and the auditor examines the processes followed and deliverables produced by each team member and assesses if there are any quality risks associated with the project based on the information provided.

The *auditor* needs good written and verbal communication skills and will consider the role that the participant is performing and relates this to the defined process for their area. The auditor writes a report detailing the findings from the audit and the recommended corrective actions with respect to any identified non-compliance to the defined procedures. He/she will perform follow-up activity at a later stage to verify that the corrective actions have been carried out. The audit activities include planning activities, the audit meeting, gathering data, reporting the findings and assigning actions, and following the actions through to closure.

1.5.4 Problem-Solving Techniques

There is a relationship between the quality of the process and the quality of the products built from the process. Defects may be due to a defect in the process itself, and so it is important to identify the causes of defects and to correct any systemic defects in the process.

Problem-solving teams are formed to solve a particular problem and to identify appropriate corrective actions. The team may be disbanded after successful resolution of the problem, and they first agree on the problem to be solved. They collect and analyse the facts and perform analysis to determine the appropriate solution. They use various tools such as fishbone diagrams, histograms, trend charts, Pareto diagrams, and bar charts to assist with problem-solving and to analyse and identify appropriate corrective actions.

Fishbone Diagrams
This well-known cause-and-effect diagram is in the shape of the backbone of a fish. The approach is to identify the possible causes of some particular quality effect. These may include people, materials, methods, and timing. Each of the main causes may then be broken down into subcauses. The root cause is then identified, as often 80% of problems are due to 20% of causes (the 80:20 rule).

Histograms
A histogram is a way of representing data via a frequency distribution in bar chart format, and it is a graphical representation of the underlying distribution of the data. It illustrates the shape, variation, and centring of the underlying distribution. The data is divided into a number of buckets, where a bucket is a particular range of data

values, and the relative frequency of each bucket is displayed in bar format. The shape of the process and its spread from the mean is evident from the histogram.

Pareto Chart
The objective of a Pareto chart is to identify the key problems and to focus on these. Problems are classified into various types or categories, and the frequency of each category of problem is then determined. The chart is displayed in a descending sequence of frequency, with the most significant category detailed first, and the least significant category detailed last. The success in problem-solving activities over a period of time may be judged from the trends in the Pareto chart, and if problem-solving activities are successful, then the key problem categories in the old chart should show a noticeable improvement in the new Pareto chart.

Trend Graph
A trend graph is a graph of a variable over time and is a study of observed data for trends or patterns over time.

Scatter Graphs
The scatter diagram is used to measure the relationship between variables and to determine whether there is a correlation between the variables. The results may be a positive correlation, negative correlation, or no correlation between the data. The scatter diagram provides a means to confirm a hypothesis that two variables are related and provides a visual means to illustrate the potential relationship.

Failure Mode Effect Analysis
This involves identifying all of the possible failures of the system and the impact of each failure. Each possible failure mode is documented, as well as the impact of failure, the cause of failure, the frequency of occurrence, its severity, the estimate of detection of the failure, the risk and corrective action to minimize the risk. FMEAs are usually applied at the design stage.

Problem-solving techniques are discussed in more detail in Chap. 9.

1.5.5 Cost of Quality

Crosby argued that the most meaningful measurement of quality is the cost of quality and that the emphasis of the improvement activities should be to reduce the *cost of poor quality* (COPQ).

The cost of quality includes the cost of external and internal failure, the cost of providing an infrastructure to prevent the occurrence of problems, and the cost of the infrastructure to verify the correctness of the product. It was divided into four subcategories (Table 1.9) by Feigenbaum in the 1950s and evolved further by James Harrington of IBM.

Table 1.9 Cost of quality categories

Type of Cost	Description
Cost external failure	This includes the cost of external failure and includes engineering repair, warranties, and a customer support function
Cost internal failure	This includes the internal failure cost and includes the cost of reworking and retesting of any defects found internally
Cost prevention	This includes the cost of maintaining a quality system to prevent the occurrence of problems and includes the cost of software quality assurance and the cost of training
Cost appraisal	This includes the cost of verifying the conformance of a product to the requirements and includes the cost of provision of software inspections and testing processes

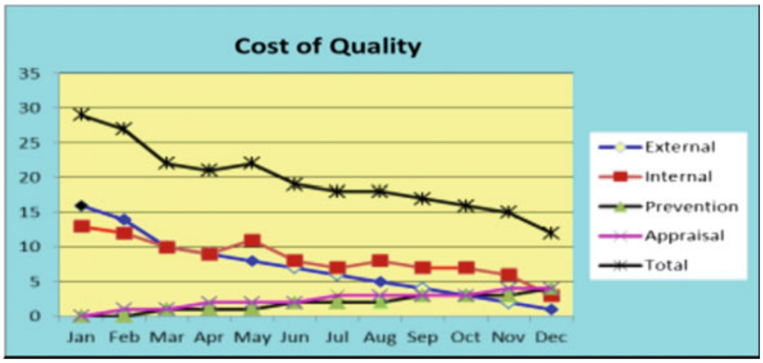


Fig. 1.9 Cost of quality

The cost of quality graph (Fig. 1.9) will initially show high external and internal costs and very low prevention costs, and the total quality costs will be high. However, as an effective quality system is put in place and becomes fully operational there will be a noticeable decrease in the external and internal cost of quality and a gradual increase in the cost of prevention and appraisal. The total cost of quality will substantially decrease, as the cost of provision of the quality system is substantially below the savings gained from lower cost of internal and external failure.

1.5.6 Software Process Improvement

Software process improvement initiatives support the organization in achieving its key business goals such as delivering software faster to the market, improving quality, reducing or eliminating waste. The objective is to work smarter and to build software better, faster, and cheaper than competitors. It makes business sense and provides a tangible return on investment.

An improvement programme is a project in its own right and needs to be managed as such. Model-based approaches to process improvement involve using models such as the CMM, CMMI, ISO 9000, PSP or TSP. A software process maturity model provides a set of best practices in software engineering, and an assessment of the organization against the model will yield the current strengths and weaknesses of the organization with respect to the model. The organization needs to prioritize the improvements that will give the greatest business return.

The employees of the company are, in effect, the owners of the process infrastructure within the organization, as they work with the processes and procedures on a daily basis. They have an interest in having the best possible processes, and a good improvement programme will empower employees to make suggestions for continuous improvement. A reward and recognition mechanism helps to make process improvement part of the organization culture.

Improvement tends to be most successful when performed in small steps rather than trying to do too much initially. It is generally easier for an organization to adjust to a series of small changes rather than one big major change. Changes within an organization need to be carefully planned and controlled. Training for the existing employees may be required to ensure that they fully understand the rationale for the proposed changes and are in a position to implement the proposed changes in the organization.

1.5.7 Software Metrics

The use of measurement is an integral part of science and engineering disciplines, and software measures are increasingly used in software engineering. The term *"software metric"* was coined by Tom Gilb in his influential book on software measurement (Gilb 1977). The purpose of measurement in software engineering is to provide an objective indication of the effectiveness of the organization in achieving its key goals and objectives.

It is essential that the measurements are relevant and closely related to the organization goal. One way to ensure this is to employ the *goal, question, metric* (GQM) approach which mandates the organization to first identify its key goals; then, it identifies the questions which need to be answered to assess the extent to which the goal is being satisfied, and then, it formulates a metric to give an objective answer to the particular question. This approach was formulated by Victor Basili and others and is described in Basili and Rombach (1988).

Measurement may be used to verify that an organization has actually improved, as quantitative data before and after the improvement initiative can be compared to judge the extent of the improvements. The initial measurements prior to the improvement programme serve as the baseline measurement of the current capability of the organization. A successful improvement programs will lead to improvements, and this will be reflected in the metrics. The implementation of metrics involves:

- Business goals
- Questions related to goals
- Metrics
- Data gathering
- Presentation of charts
- Trends
- Action plans.

Metrics are discussed in more detail in Chap. 9.

1.5.8 Customer Satisfaction

The customer will ultimately judge the effectiveness of the quality management system in delivering high-quality software, and the level of customer satisfaction will influence the customer in purchasing again from the company or recommending the company. Customer satisfaction surveys are used to determine the level of customer satisfaction with the company.

A customer satisfaction survey involves the customer rating the organization in several key areas such as the quality of the software, its reliability, the timeliness of delivery, and so on. The process takes the form of a closed feedback loop, and the customer satisfaction feedback will be analysed and acted upon appropriately.

The survey is conducted, and the feedback analysed and used to prepare the action plan. The actions are executed, and the customer is surveyed again at later date (Fig. 1.10). The follow-up activity may involve a telephone conversation with the customer or a visit to the customer site to discuss the specific issues. The objective is to ensure that customers are totally satisfied with the product and service, as a loyal customer will repurchase and recommend the company to other potential customers.

Fig. 1.10 Customer satisfaction process

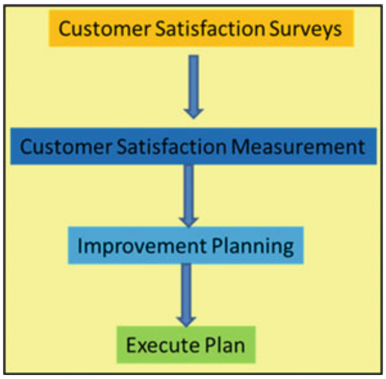

Customer Satisfaction Surveys
Customer Satisfaction Measurement
Improvement Planning
Execute Plan

Table 1.10 Sample customer satisfaction questionnaire

No	Question	Unacceptable	Poor	Fair	Satisfied	Excellent	N/A
1.	Quality of software	☐	☐	☐	☐	☐	☐
2.	Ability to meet agreed dates	☐	☐	☐	☐	☐	☐
3.	Timeliness of projects	☐	☐	☐	☐	☐	☐
4.	Effective testing of software	☐	☐	☐	☐	☐	☐
5.	Expertise of staff	☐	☐	☐	☐	☐	☐
6.	Value for money	☐	☐	☐	☐	☐	☐
7.	Quality of support	☐	☐	☐	☐	☐	☐
8.	Ease of installation	☐	☐	☐	☐	☐	☐
9.	Ease of use	☐	☐	☐	☐	☐	☐
10.	Timely problem resolution	☐	☐	☐	☐	☐	☐

The customer satisfaction process is summarized as follows:

- Define customer surveys
- Send customer surveys
- Customer satisfaction ratings
- Customer meeting and key issues
- Action plans and follow-up
- Metrics for customer satisfaction.

The questionnaire will vary according to the business, but it will cover the relevant questions to determine where the organization is weak and areas where it is strong. The questions typically employ a rating scheme to allow the customer to give quantitative feedback on satisfaction, and the survey will also enable the customer to go into more detail on issues.

A sample survey form with 10 questions is included in Table 1.10, and the form will also include open-ended questions to enable the customer to go into more detail on any issues. Customer satisfaction metrics provide visibility into the level of customer satisfaction and enable trends to be determined (Fig. 1.11).

1.5.9 Assessments (Appraisals)

The objective of an assessment (or *appraisal*) of an organization is to determine its maturity with respect to a maturity model such as the CMMI or SPICE or against an international quality standard such as ISO 9000.

The appraisal is performed by an external or internal assessment team and yields the strengths and weaknesses of the organization with respect to the model. The appraisal report is used to plan and prioritize future improvements.

It is a major review of the organization, and it needs to be conducted by an experienced assessment team. It involves interviews with the project managers and

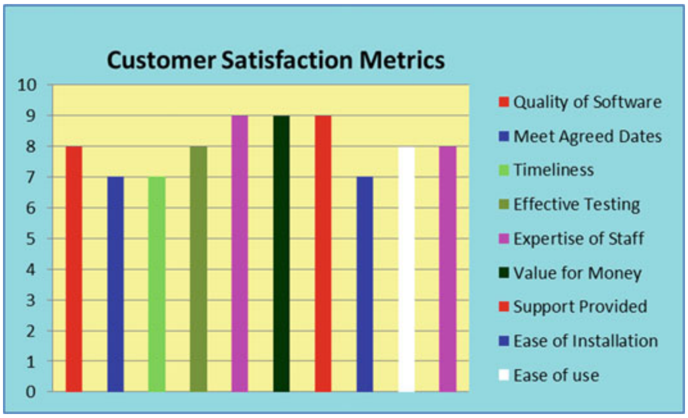


Fig. 1.11 Customer satisfaction metrics

project teams as well as the review of relevant documentation. The assessment report will detail the extent to which the model is implemented, and any gaps and improvement opportunities are highlighted in the report.

1.5.10 Total Quality Management

Total quality management (TQM) is a management philosophy that is focused on quality and on developing a culture of quality in the organization. It is a holistic approach, and it applies to all levels and functions within the organization. Quality is a company-wide objective, and the goal is total customer satisfaction. The company aims to deliver products and services that totally satisfy the customer needs.

TQM uses many of the ideas of the pioneers in the quality movement. Management is required to take charge of the implementation of quality management, and all staff will need to be trained in quality improvement activities.

The implementation of TQM involves a focus on all areas in the organization and in identifying potential improvements. The problems in the particular area are evaluated, and data is collected and analysed. An action plan is then prepared and the actions implemented and monitored. This is then repeated for continuous improvement. The implementation is summarized as follows:

- Identify improvement area
- Problem evaluation
- Data collection
- Data analysis
- Action plan
- Implementation of actions

Table 1.11 Total quality management

Part	Description
Customer focus	This involves identifying internal and external customers and recognizing that all customers have expectations and rights which need to be satisfied first time and every time. Quality must be considered in every aspect of the business, and the focus is on fire prevention
Process	This involves a focus on the process and improvement to the process via problem-solving to reduce waste and eliminate errors
Measurement and analysis	This involves setting up a measurement programme to enable objective and effective analysis of the quality of the process and product
Human factors	This involves developing a culture of quality and customer satisfaction throughout the organization. The core values of quality and customer satisfaction need to be instilled in the organization. This requires training for the employees on quality, customer satisfaction, and continuous improvement

- Monitor effectiveness
- Repeat.

There are four main parts of TQM which are summarized in Table 1.11.

The ISO 9000 standard [see Chap. 11 of O'Regan (2014)] is a structured approach to the implementation of TQM. Its clauses are guidelines for what needs to be done and include requirements to be satisfied for the organization to satisfy ISO 9000.

1.6 Miscellaneous

Software quality management is, in many ways, the application of common sense to software engineering. In this section, we discuss organization culture and change as well as legal aspects of failure, and finally, we discuss quality and the Web.

1.6.1 Organization Culture and Change

Every organization has a distinct culture, and this reflects the way in which things are done in the company. Organization culture includes the ethos of the organization, its core values, its history, its success stories, its people, amusing incidents, and so on. The culture of the organization may be favourable or unfavourable to developing high-quality software.

Occasionally, a change to the organization culture is required, and this may be difficult as it could involve changing its fundamental ways of working, and there may be a resistance to this. Successful change management often involves:

- Kick-off meeting
- Motivate rationale for changes
- Present plan
- Training
- Implement changes
- Monitor implementation
- Institutionalize.

The culture of an organization is often illustrated by the phrase: *"That's the way we do things around here"*. For example, the evolution from one level of the CMM to another often involves a change the way that things are done in the organization. The focus on prevention requires a change in mindset to focus on *problem-solving* and *fire prevention*, rather than on *firefighting*.

1.6.2 Law of Negligence

The impact of a flaw in software may be catastrophic, and several software failures were discussed earlier in this chapter. Clearly, every organization must take all reasonable precautions to prevent the occurrence of defects, especially in the safety-critical domain where defects may cause major damage or even loss of life. Reasonable precautions consist of having appropriate software engineering practices in place to allow the organization to consistently produce high-quality software.

A quality management system indicates that the organization takes software quality seriously and that has a sound software development process in place that serves the needs of the organization and its customers. Modem quality assurance systems include processes for software inspections, testing, quality audits, customer satisfaction, software development, project planning, etc.

The organization will require evidence or records to prove that the quality management system is in place that it is appropriate for the organization and that it is fully operational within the organization. The proof that the quality system is actually operational typically takes the form of records of the various activities. The records also enable the organization to prepare a legal defence to show that it took all reasonable precautions in software development, especially if a customer decides to take legal action for negligence against the software provider following a serious problem in the software at the customer site.

The presence of records may be used to indicate that all reasonable steps were taken, and the records typically include lists of all the deliverables in the project; minutes of project meetings; records of reviews of requirements, design, and software code, records of test plans and test results; and so on.

1.6.3 Quality and the Web

The explosive growth of the World Wide Web and electronic commerce has made the quality of websites a key concern. Web technology is rapidly becoming ubiquitous in society and is quite distinct from other software systems in that:

- It may be accessed from anywhere in the world
- It may be accessed by many different browsers
- The usability and look and feel of the application is a key concern
- The performance of the website is a key concern
- Security is a key concern
- The website must be capable of dealing with a large number of transactions at any time
- The website has very strict availability constraints (typically 24 × 365)
- The website needs to be highly reliable.

It is inappropriate to employ the waterfall lifecycle for this domain, and usually a spiral lifecycle will be employed as the requirements are often incomplete at project initiation and evolve to the agreed set during the project. Often, rapid application development (RAD), joint application development (JAD) or the Agile methodology is employed.

1.7 Review Questions

1. Discuss the contributions of Deming and Juran.
2. Describe Crosby's maturity grid and discuss how it influenced the Capability Maturity Model?
3. Explain why Watts Humphrey is considered the father of software quality.
4. Explain the difference between software inspections and testing?
5. What is an assessment (appraisal) and explain how it forms part of the improvement cycle.
6. Why is the cost of poor quality an important measure?
7. Discuss the role of software metrics in problem-solving.
8. Explain the importance of customer satisfaction and describe how it may be measured.

1.8 Summary

This chapter gave a short introduction to the software quality field, and we discussed the contributions of several pioneers in the quality field including Shewhart, Deming, Juran, and Crosby. We also discussed Watts Humphrey, who is considered the father of software quality.

We examined various definitions of quality such as Crosby's "conformance to the requirements" and Juran's "fitness for purpose", as well as considering the various dimensions of software product quality listed in ISO 9126.

We considered several software failures such as the Ariane 5 disaster, the year 2000 problem, and a maths bug in the Intel Pentium microprocessor. A software failure may have devastating consequences and so it is essential to develop high-quality software.

We discussed software inspections that build quality into the software; software testing that verifies that the software is of high quality as well as finding defects in the software; software quality assurance to provide visibility into the processes; problem-solving techniques to prevent problems from re-occurring; the cost of poor quality to the organization; software process improvement to improve the key processes in the organization; and customer satisfaction to determine the level of customer satisfaction with the organization.

References

Basili V, Rombach H (1988) The TAME project. Towards improvement-oriented software environments. IEEE Trans Softw Eng 14(6)
Brooks F (1975) The mythical man month. Addison Wesley, Boston
Brooks F (1986) No silver bullet. Essence and accidents of Software Engineering. Information processing. Elsevier, Amsterdam
Crosby P (1979) Quality is free. The art of making quality certain. McGraw Hill, New York
Deming WE (1986) Out of crisis. M.I.T. Press, Cambridge
Fagan M (1976) Design and code inspections to reduce errors in software development. IBM Syst J 15(3)
Gilb T (1977) Software metrics. Winthrop Publishers, Winthrop
Humphry W (1989) Managing the software process. Addison Wesley, Boston
ISO/IEC 9126 (1991) Information Technology. Software product evaluation: quality characteristics and guidelines for their use
Juran J (1951) Juran's quality handbook. McGraw Hill, New York
Lions JL (1996) Ariane 5. Flight 501. Failure report by enquiry board
Manley E (1995) Taurus: how I lived to tell the tale (American Programmer: Software failures)
O'Regan G (2014) Introduction to software quality. Springer, Berlin
Shewhart W (1931) The economic control of manufactured products. D. van Nostrand & Co. Inc., New York
Standish Group Research Note (1999) Estimating: art or science. Featuring Morotz cost expert