



# Certified Software Quality Assurance Professional Sample Material

## V-Skills Certifications

A Government of India  
&  
Government of NCT Delhi Initiative

V-Skills



## 1. INTRODUCTION


Software quality assurance (SQA) consists of a means of monitoring the software engineering processes and methods used to ensure quality. The methods by which this is accomplished are many and varied, and may include ensuring conformance to one or more standards, such as ISO 9000 or a model such as CMMI.

SQA encompasses the entire software development process, which includes processes such as requirements definition, software design, coding, source code control, code reviews, software configuration management, testing, release management, and product integration. SQA is organized into goals, commitments, abilities, activities, measurements, and verifications.

### 1.1. History of Quality Movement

The quality movement can trace its roots back to medieval Europe, where craftsmen began organizing into unions called guilds in the late 13th century. Until the early 19th century, manufacturing in the industrialized world tended to follow this craftsmanship model. The factory system, with its emphasis on product inspection, started in Great Britain in the mid-1750s and grew into the Industrial Revolution in the early 1800s. In the early 20th century, manufacturers began to include quality processes in quality practices. After the United States entered World War II, quality became a critical component of the war effort: Bullets manufactured in one state, for example, had to work consistently in rifles made in another. The armed forces initially inspected virtually every unit of product; then to simplify and speed up this process without compromising safety, the military began to use sampling techniques for inspection, aided by the publication of military-specification standards and training courses in Walter Shepherd's statistical process control techniques.

In the few years since the turn of the century, the quality movement seems to have matured beyond Total Quality. New quality systems have evolved from the foundations of Deming, Juran and the early Japanese practitioners of quality, and quality has moved beyond manufacturing into service, healthcare, education and government sectors.

TIME:	Early 1900s	1940s	1960s	1980s and Beyond
FOCUS:	Inspection	Statistical sampling	Organizational quality focus	Customer driven quality
 <p>Old Concept of Quality: Inspect for quality after production.</p>				 <p>New Concept of Quality: Build quality into the process. Identify and correct causes of quality problems.</p>

Evolution of Quality Management

## 1.2. Continuous Improvement

Continuous improvement involves constantly identifying and eliminating the causes that prevent a system or process from functioning at its optimum level. The concept of continuous improvement originated in Japan in the 1970s. It was adopted in many countries, including U.S.A., in the early 1980s. Continuous improvement—and consequent customer satisfaction—is the principle on which the concept of Lean manufacturing is developed. When this principle is combined with just-in-time technique, it results to Lean manufacturing. Continuous improvement helps an organization to add value to its products and services by reducing defects, mistakes, etc. and to maximize its potential. As continuous improvement requires constant ongoing efforts, it is essential that the top management takes a long term view and commits itself for its implementation.

Continuous improvement enables organizations identify and rectify problems as and when they occur. Thus, it ensures smooth functioning of the processes. Many modern quality improvement models or tools like control charts, sampling methods, process capability measures, value analysis, design of experiments, etc. have been influenced by the concept of continuous improvement.

## 1.3. Quality Pioneers

Various pioneers emerged who helped shape quality principles and laid the foundations for six sigma. They included

Walter A. Shewhart - He is the pioneer of Modern Quality Control who, recognized the need to separate variation into assignable and un-assignable causes. He is the founder of the control chart and originator of the plan-do-check-act cycle. He was the first to successfully integrate statistics, engineering, and economics and defined quality in terms of objective and subjective quality.

Dr. W. Edwards Deming - He studied under Shewhart at Bell Laboratories and major contributions includes developing 14 points on Quality Management, a core concept on implementing total quality management, is a set of management practices to help companies increase their quality and productivity. The 14 points are

- ✓ Create constancy of purpose for improving products and services.
- ✓ Adopt the new philosophy.
- ✓ Cease dependence on inspection to achieve quality.
- ✓ End the practice of awarding business on price alone; instead, minimize total cost by working with a single supplier.
- ✓ Improve constantly and forever every process for planning, production and service.
- ✓ Institute training on the job.
- ✓ Adopt and institute leadership.
- ✓ Drive out fear.
- ✓ Break down barriers between staff areas.
- ✓ Eliminate slogans, exhortations and targets for the workforce.
- ✓ Eliminate numerical quotas for the workforce and numerical goals for management.
- ✓ Remove barriers that rob people of pride of workmanship, and eliminate the annual rating or merit system.
- ✓ Institute a vigorous program of education and self-improvement for everyone.
- ✓ Put everybody in the company to work accomplishing the transformation.

Joseph Juran - His major contributions are directing most of his work at executives and the field of quality management and developing the “Juran Trilogy” for managing quality, as Quality planning, quality control, and quality improvement. He also enlightened the world on the concept of the “vital few, trivial many” which is the foundation of Pareto charts.

Philip Crosby - He stressed on Quality management and four absolutes of quality including

- ✓ Quality is defined by conformance to requirements.
- ✓ System for causing quality is prevention not appraisal.
- ✓ Performance standards of zero defects not close enough.
- ✓ Measurement of quality is the cost of nonconformance.

Arman Feigenbaum - He developed a systems approach to quality (all organizations must be focused on quality) by emphasizing that costs of quality may be separated into costs for prevention, appraisal, and failures (scrap, warranty, etc.)

Kaoru Ishikawa - He developed the concept of true and substitute quality characteristics as

- ✓ True characteristics are the customer’s view
- ✓ Substitute characteristics are the producer’s view
- ✓ Degree of match between true and substitute ultimately determines customer satisfaction

He also advocated of the use of the 7 tools and advanced the use of quality circles or worker quality teams. He also developed the concept of Japanese Total Quality Control

- ✓ Quality first and not short term profits.
- ✓ Next process is the customer.
- ✓ Use facts and data to make presentations.
- ✓ Respect for humanity as a management philosophy of full participation

Genichi Taguchi - He developed the quality loss function (deviation from target is a loss to society) and promoted the use of parameter design (application of Design of experiments) or robust engineering. The goal is to develop products and processes that perform on target with smallest variation insensitive to environmental conditions and the focus is on engineering the design.

#### **1.4. Quality Management in Software**

Software can be defined as Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system. Software is a special product as it is invisible, there are limited opportunities to detect defects, new functionalities are required to be added frequently and it has to realise high complexity.

These factors make a systematic quality assurance process a necessity, to control costs and avoid re-work. Software Quality can be defined as the degree to which a system, component, or process meets specified requirements.

Later the defect or bug is identified in the software development life cycle, higher are the costs and rework. Some issues faced in Quality Assurance of Software are

- ✓ Conflict between customer quality requirements (efficiency, reliability) and developer quality requirements (maintainability, reusability)
- ✓ Some quality requirements are difficult to specify in an unambiguous way
- ✓ Software specifications are usually incomplete and often inconsistent

### 1.5. Quality Concepts

Various quality related terms are discussed.

#### **Quality Defined**

It is defined as characteristic or attribute of a product or service. Refers to measurable characteristics that we can compare to known standards, in software it involves such measures as cyclomatic complexity, cohesion, coupling, function points, and source lines of code. It includes variation control

- ✓ A software development organization should strive to minimize the variation between the predicted and the actual values for cost, schedule, and resources
- ✓ They should make sure their testing program covers a known percentage of the software from one release to another
- ✓ One goal is to ensure that the variance in the number of bugs is also minimized from one release to another

There are two major types of quality

- ✓ **Quality of Design:** Design quality refers to the level of characteristics that the designers specify for a product.
- ✓ **Quality of Conformance:** The degree to which the design specifications are followed during manufacturing, this focuses on how well the implementation follows the design and how well the resulting system meets its requirements

#### **Quality Control**

It involves a series of inspections, reviews, and tests used throughout the software process, ensures that each work product meets the requirements placed on it. Also, includes a feedback loop to the process that created the work product which is essential in minimizing the errors produced.

Combines measurement and feedback in order to adjust the process when product specifications are not met; requires all work products to have defined, measurable specifications to which practitioners may compare to the output of each process.

#### **Quality Management**

It serves as an umbrella activity that is applied throughout the software process, involves doing the software development correctly versus doing it over again, it reduces the amount of rework which results in lower costs and improved time to market. It encompasses

- ✓ A software quality assurance process
- ✓ Specific quality assurance and quality control tasks (including formal technical reviews and a multi-tiered testing strategy)
- ✓ Effective software engineering practices (methods and tools)
- ✓ Control of all software work products and the changes made to them
- ✓ A procedure to ensure compliance with software development standards
- ✓ Measurement and reporting mechanisms

### Quality Assurance Functions

It consists of a set of auditing and reporting functions that assess the effectiveness and completeness of quality control activities; provides management personnel with data that provides insight into the quality of the products; alerts management personnel to quality problems so that they can apply the necessary resources to resolve quality issues.

### Product

A product as any tangible output or service that is a result of a process. A product itself may include hardware, software, documentation, or a combination of these; also note that a service is included in the definition.

### Process

A process is a set of activities performed for a given purpose, for example, a software acquisition process. The quality of a product or service is dependent on the quality of the process used to create it.

### Requirement

A requirement is a needed capability, condition, or a property that must be possessed by an entity to satisfy a contract, standard, specification, or other formally imposed documents. A requirement is a way of characterizing a user's need in a way that allows a development team to implement that need in the product in a concrete way. The achievement of the requirement is the yardstick by which we measure the quality of a product or a service.

### User

A user is either the customer or the end user. The user is either the customer (either internal or external) or represented by a buyer or the user community.

### Evaluation

It is defined as the process of determining satisfaction of requirements. As defined by Kenett is defines evaluation as "a process to evaluate the quality of products and to evaluate associated documentation, processes, and activities that impact the quality of the products". Evaluation includes methods such as analyses, inspections, reviews, and tests.

### Measure and Measurement

Measure is to ascertain the characteristics or features (extent, dimension, quantity, capacity, and capability) of something, especially by comparing with a standard. Measurement is a dimension, capacity, quantity, or amount of something (e.g., 300 source lines of code or seven document pages

of design). Measurement is the process by which numbers or symbols are assigned to attributes of entities in the real world in such a way as to characterize the attributes by clearly defined rules.

### Failure

A failure is said to occur whenever the external behavior of a system does not conform to that prescribed in the system specification.

### Error

An error is a state of the system. In the absence of any corrective action by the system, an error state could lead to a failure which would not be attributed to any event subsequent to the error.

### Fault

A fault is the adjudged cause of an error.

### Cost of Quality

The "cost of quality" isn't the price of creating a quality product or service. It's the cost of not creating a quality product or service. Every time work is redone, the cost of quality increases. Cost of quality is the sum of various costs as that of appraisal costs, prevention costs, external failure costs, and internal failure costs. It is generally believed that investing in prevention of failure will decrease the cost of quality as failure costs and appraisal costs will be reduced. Understanding cost of quality helps organizations to develop quality conformance as a useful strategic business tool that improves their product, services & brand image. This is vital in achieving the objectives of a successful organisation.

COQ is primarily used to understand, analyze & improve the quality performance. COQ can be used by shop floor personnel as well as a management measure. It can also be used as a standard measure to study an organization's performance vis-à-vis another similar organisation and can be used as a benchmarking indices.

Various types of Quality Costs are

**Appraisal Costs:** The costs associated with measuring, evaluating or auditing products or services to assure conformance to quality standards and performance requirements. These include the costs of

- ✓ Inspection of software requirements
- ✓ In-process and final inspection
- ✓ Product and process audits

**Prevention costs:** Prevention Costs are any costs that are incurred in an effort to minimize appraisal and failure costs. This category is where most quality professionals want to live. They say an ounce of prevention is worth a pound of cure and they are what this category is all about. This includes the activities that contribute to creation of the overall quality plan and the numerous specialized plans. Examples are costs associated with Quality planning, formal technical reviews, test equipment, training.

**Failure costs:** The costs resulting from products or services not conforming to requirements or customer/user needs. Failure costs are divided into internal and external failure categories.

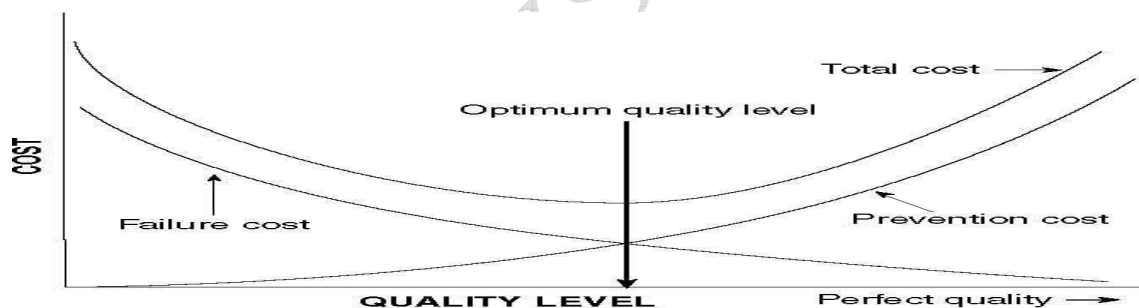
- ✓ Internal failure costs: Incurred when an error is detected in a product prior to shipment; Include rework, repair, and failure mode analysis
- ✓ External failure costs: Involves defects found after the product has been shipped
- ✓ Include complaint resolution, product return and replacement, help line support, and warranty work.

Examples of the various costs are

- ✓ Prevention - Training Programme, Preventive Maintenance
- ✓ Appraisal - Depreciation of Test/ Measuring Equipment, Inspection Contracts
- ✓ Internal Failure - Scrap, Rework, Downtime, Overtime
- ✓ External Failure - Warranty, Allowances, Customer Returns, Customer Complaints, Product Liability, Lawsuits, Lost Sales

Identifying COQ can have several benefits, as

- ✓ It provides a standard measure across the organisation & also inter-organisation
- ✓ It builds awareness of the importance of quality
- ✓ It identifies improvement opportunities
- ✓ Being a cost measure, it is useful at shop floor as well as at management level



### *Factors for Poor Quality*

Some of the factors which result in poor quality software are

- ✓ Lack of domain knowledge: Most developers are not experts in the business domain served by their applications, be it telecommunications, banking, energy, supply chain, retail, or others. The best way to mitigate this is to provide access to domain experts from the business, proactively train developers in the business domain, and conduct peer reviews with those possessing more domain experience.
- ✓ Lack of technology knowledge: Most developers are proficient in several computer languages and technologies. However, modern multi-tier business applications are a complex tangle of many computer languages and different software platforms. These tiers include user interface, business logic, and data management, and they may interact through middleware with enterprise resource systems and legacy applications written in archaic languages. Few



developers know all of these languages and technologies, and their incorrect assumptions about how other technologies work is a prime source of the non-functional defects that cause damaging outages, data corruption, and security breaches during operation. The best way to mitigate this cause is to cross-train developers in different application technologies, conduct peer reviews with developers working in other application tiers, and perform static and dynamic analyses of the code.

- ✓ **Badly engineered software:** Two-thirds or more of most software development activity involves changing or enhancing existing code. Unnecessarily complex code is often impenetrable and modifying it leads to numerous mistakes and unanticipated negative side effects. These newly injected defects cause expensive rework and delayed releases. The best way to mitigate this cause is to re-factor critical portions of the code guided by information from architectural and static code analyses.
- ✓ **Unrealistic schedules:** When working at breakneck pace, stressed developers make more mistakes and have less time to find them. The only way to mitigate these death march travesties is through enforcing strong project management practices. Controlling commitments through planning, tracking progress to identify problems, and controlling endless requirements changes are critical practices for providing a professional environment for software development.

### *Customer View of Quality*

From the customer's perspective, satisfaction after the delivering of the product is the ultimate validation of the product quality. It is clear that the concept of quality must involve customers or, simply put, quality is conformance to customers' expectations and requirements

Customer views the software as a quality product if it satisfies the below mentioned criteria

- ✓ The product received is able to perform the task for which it was purchased
- ✓ All the requirements and the needs are being met by the product
- ✓ During the transaction they received a treatment which maintained their integrity and respect.

### **1.6. Customer Expectations**

Advancements in software design and deployment strategy have shaped customer's expectations about what quality software ought to be like. Expectations are so high, many argue that there is a whole range of software products that will not be developed because they cannot be made profitable. The less-depressing counter-point is that high customer expectations demand thoughtfulness from designers and excellence from developers – both good things. Some of the customer expectations are

- ✓ **User-centric design** - Software must be easy for everyone to use, not just tech-savvy power users. If a 13 year old cannot easily navigate through your software while simultaneously surfing Facebook, watching YouTube and texting, it's too complicated. Successful software in the 2010's is easy to use.
- ✓ **Instant Availability** - Customers demand to hear about, try for the first time, and understand the software immediately. A "coming soon" message takes a lot of trust capital that most young software companies just do not have.
- ✓ **Universal Availability** - The cloud is here to stay. Customers are annoyed if software asks them to download/install/update. Those things should happen automatically, invisibly, and software should jump to a new device as seamlessly as the customers.

- ✓ Clarity - Providing partial assistance with things you are not clear about is not a good practise. If customers face some difficulty with the software, they should be directed towards the right team, and not provided incomplete information from some non relevant team.
- ✓ Customers should be kept in loop - Customers don't like to be kept in dark, they should be constantly updated about the developments taking place in the process and in case of resolving an issue and they should be provided timely information about the work being done to resolve the issue.

www.vskills.in