|  |
| --- |
| **Objectives:**   * **To familiarize students with the basic concepts in quality** * **To show what parameters are included while defining software quality** * **To give idea about quality concepts like TQM, ISO, SEI- CMM etc.** |

**2.1 Definitions of Quality**

Quality is defined as meeting the customer’s requirements in the first time and every time. Quality is much more than the absence of defects, which allows us to meet customer’s expectations.

**2.1.1 Parameters of Software Quality**

Quality of software can be expressed by a collection of parameters that may include:

* Adequacies of fulfillment of all requirement specifications- functional, behavioral, performance and interface etc.
* Reliability
* Robustness and fault/ failure tolerance
* Maintainability (good documentation and process standards play a vital role here)
* User-friendliness
* Usability
* Learn-ability
* Resource utilization efficiency
* QoS (quality of service) levels delivered and so on….

Generally speaking, quality software should fulfill all its basic requirements, and be reasonably bug-free, delivered on time and within budget, and maintainable.

However, quality is obviously a subjective term that entirely depends on relative judgments and interpretations from different viewpoints of the stakeholders involved e.g. the developer, the tester, the auditor, the customer and the market in general.

For example, it will depend a lot on who the 'customer' is and their overall influence in the scheme of things. Customers of a software development project might include end-users, customer acceptance testers, customer contract officers, customer management, the development organization's management / accountants / testers / salespeople, future software maintenance engineers, stockholders, magazine columnists.

Patrick Town send examines quality in fact and quality in perception. Quality in perceptions is the customer’s point of view and quality in fact is supplier’s point of view.

**2.1.2 Quality in Fact**

**Customer‘s View**

* Doing the right thing
* Doing the right way
* Doing it right first time
* Doing it on time

**2.1.3 Quality in Perception**

**Supplier‘s View**

* Delivering the right product
* Satisfying the customer’s need
* Meeting the customer’s expectation
* Treating every customer with integrity, courtesy, and respect

**2.2 Why we need a Quality?**

Quality is the most important factor affecting an organization’s long term performance. Quality is the way to achieve improved productivity and competitiveness in any organization. Quality saves, it does not cost. Quality is the solution to the problem, not a problem.

**2.3 Factors influencing the Project**

The following are the factors influencing the project:

* Quality – Always Expect the Quality Product
* Cost – Should be a Reasonable Cost
* Schedule –Should be completed on time

**2.4 Cost of Quality**

When calculating the total production costs associated with the development of a new application or system, two cost components must be considered.

The first component consists of costs associated with producing the product “right the first time” or RTF costs. These costs include labor, materials, and tools associated the actual development of the application. Cost of quality is a term that is used to quantify the total cost of failure, appraisal, and prevention costs associated with the production of software. This cost component is called the cost of quality.

The three categories of costs associated with producing quality products are:

**a) Prevention Cost:** Money required to prevent errors and to do the job right for first time. This category includes money spent on establishing methods and producers, training workers, acquiring tools and planning for quality. Prevention of money is all spent before the product is actually built.

**b) Appraisal Cost:** Money spent to review completed product against requirement. Appraisal includes the cost of inspection, testing, and reviews. This money spent after the product is built but before it is shipped to the user.

**c) Failure Cost:** This cost is associated with defective products that have been delivered to the user or moved into production. Some failure costs involve repairing products to make them fit as per requirements.

Cost of Production = Right the First Time Cost (RTF) + Cost of Quality

RTF Cost = Cost of labor, material, tool etc

Cost of Quality = Prevention Cost + Appraisal Cost + Failure Cost

**2.5 Quality Team**

Quality team can be classified into two types:

* Quality Assurance
* Quality Control

Quality Team

Quality Control

Quality Assurance

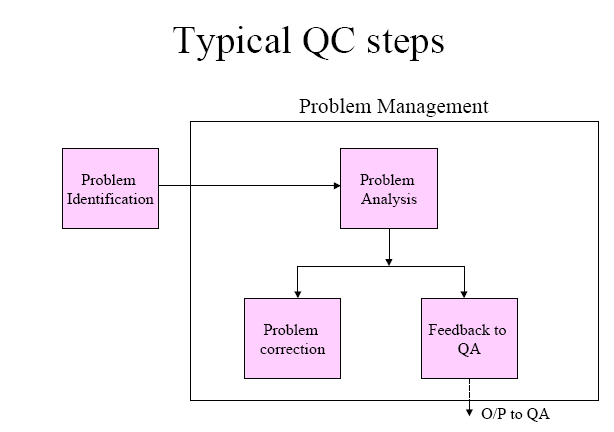
**Fig 2.1 Quality Team**

**Quality Assurance** is a planned and systematic set of activities necessary to provide adequate confidence that product and services will conform to specified requirements and meet user needs. Quality assurance is staff function, responsible for implementing the quality policy defined through the development and continuous improvement of software development process. It is process oriented. It is defect prevention based.

**NOTE:** Both Developer and Software Tester comprises Quality Assurance

**Quality Control** is the process by which product quality is compared with applicable standards and the action taken when non-conformance is detected. Quality Control is line function, and work is done within a process to ensure that work product conforms to standards and/or requirements. It is product based. Also it is defect detection based.

**NOTE:** People can be called as a 100 % tester. E.g., ISO, ISI

****

**ig 2.2 Quality control Steps**

**Quality Control Vs Quality Assurance**

|  |  |
| --- | --- |
| **QC** | **QA** |
| Product | Process |
| Reactive | Proactive |
| Line Function | Staff Function |
| Find Defects | Prevent Defects |

**QC Vs QA – Examples**

|  |  |
| --- | --- |
| **QC** | **QA** |
| Walkthrough | Quality Audit |
| Testing | Defining Process |
| Inspection | Selection of Tools |
| Checkpoint Review | Training |

**2.6 Software Quality Systems**

**SEI-CMM** = (process-oriented, organizational-level applications)

* **SEI** = 'Software Engineering Institute' at Carnegie-Mellon University; initiated by the U.S. Defense Department to help improve software development processes. The SEI publishes books on [software engineering](http://en.wikipedia.org/wiki/Software_engineering) for industry, government and military applications and practices.
* **CMM** = Capability Maturity Model (CMM) broadly refers to a [process improvement](http://en.wikipedia.org/wiki/Process_improvement) approach that is based on a [process](http://en.wikipedia.org/wiki/Process) model. CMM also refers specifically to the first such model, developed by the [Software Engineering Institute](http://en.wikipedia.org/wiki/Software_Engineering_Institute) (SEI) in the mid-1980s, as well as the family of process models that followed.

The Capability Maturity Model can be used to assess an organization against a scale of five process maturity levels. Each level ranks the organization according to its standardization of processes in the subject area being assessed. The subject areas can be as diverse as software engineering, systems engineering, project management, risk management, system acquisition, [information technology](http://en.wikipedia.org/wiki/Information_technology) (IT) services and personnel management.

'Capability Maturity Model', variations:

* + **PCMM** (People-Capability Maturity Model)
  + **CMMI** (Capability Maturity Model Integration)

It is a quality model with 5 levels of process 'maturity' that determine effectiveness in delivering quality software. Organizations can receive CMMI ratings by undergoing assessments by qualified auditors.

The five levels are broadly described below:

**Level 1 – Initial** 🡪 characterized by chaos, periodic panics, and heroic efforts required by individuals to successfully complete projects. Few if any processes in place; successes may not be repeatable.

**Level 2 – Repeatable** 🡪 characterized by project level thinking and it is reactive. It includes software project tracking, requirements management, realistic planning and configuration management

**Level 3 – Defined** 🡪 characterized by organizational level thinking and it is proactive. It includes standardized processes for software development and maintenance processes that are integrated throughout an organization; a Software Engineering Process Group is in place to oversee, and training programs are used to ensure understanding and compliance.

**Level 4 – Managed** 🡪 characterized by controlled processes. It includes measurable processes- metrics are used to track productivity, processes and products. Project performance is predictable and quality is consistently high.

**Level 5 – Optimizing** 🡪 continuous process improvements. The impact of new processes and technologies can be predicted and effectively implemented when required.

During 1997-2001, 1018 organizations were assessed. Of those, 27% were rated at Level 1, 39% at 2, 23% at 3, 6% at 4 and 5% at 5. For those rated at Level 1, the most problematical key process area was in Software Quality Assurance.

**2.6.1 ISO (International Organization for Standardization) - The ISO 9001:2000 standard**

This model focuses on quality systems that are assessed by outside auditors. Originating from traditional manufacturing organizations, and thereby getting extended to the services domain, it applies to many kinds of organizations, including software. The uniqueness of this model is that ISO certification does not necessarily indicate quality products - it only indicates that standardized and documented processes exist and are followed. For ISO 9001 certification, a third-party auditor assesses an organization, and certification is typically good for about 3 years, after which a complete reassessment is required. It covers documentation, design, development, production, testing, installation, servicing, and other processes. The full set of standards consists of:

**Q9001-2000** - Quality Management Systems: Requirements;

**Q9000-2000** - Quality Management Systems: Fundamentals and Vocabulary;

**Q9004-2000** - Quality Management Systems: Guidelines for Performance

improvements.

**2.6.2 ISO 9126 standard**

ISO 9126 is an [international standard](http://en.wikipedia.org/wiki/International_standard) for the [evaluation](http://en.wikipedia.org/wiki/Evaluation) of [software](http://en.wikipedia.org/wiki/Software). The standard is divided into four parts which addresses, respectively, the following subjects: quality model; external metrics; internal metrics; and quality in use metrics.

The quality model established in the first part of the standard, ISO 9126-1, classifies [software quality](http://en.wikipedia.org/wiki/Software_quality) in a structured set of characteristics and sub-characteristics as follows:

* **Functionality** - A set of attributes that bear on the existence of a set of functions and their specified properties. The functions are those that satisfy stated or implied needs.
  + Suitability
  + Accuracy
  + [Interoperability](http://en.wikipedia.org/wiki/Interoperability)
  + Compliance
  + [Security](http://en.wikipedia.org/wiki/Computer_security)
* [**Reliability**](http://en.wikipedia.org/wiki/Reliability) - A set of attributes that bear on the capability of software to maintain its level of performance under stated conditions for a stated period of time.
  + Maturity
  + Recoverability
  + [Fault Tolerance](http://en.wikipedia.org/wiki/Fault_Tolerance)
* [**Usability**](http://en.wikipedia.org/wiki/Usability) - A set of attributes that bear on the effort needed for use, and on the individual assessment of such use, by a stated or implied set of users.
  + Learnability
  + Understandability
  + [Operability](http://en.wikipedia.org/wiki/Operability)
* [**Efficiency**](http://en.wikipedia.org/wiki/Algorithmic_efficiency) - A set of attributes that bear on the relationship between the level of performance of the software and the amount of resources used, under stated conditions.
  + Time Behavior
  + Resource Behavior
* [**Maintainability**](http://en.wikipedia.org/wiki/Maintainability) - A set of attributes that bear on the effort needed to make specified modifications.
  + Stability
  + Analysability
  + Changeability
  + [Testability](http://en.wikipedia.org/wiki/Testability)
* [**Portability**](http://en.wikipedia.org/wiki/Software_portability) - A set of attributes that bear on the ability of software to be transferred from one environment to another.
  + Installability
  + Replaceability
  + Adaptability

**2.6.3 IEEE (Institute of Electrical and Electronics Engineers)**

This apex body for computing has created standards such as

* IEEE Standard for Software Test Documentation - (IEEE/ANSI Standard 829),
* IEEE Standard of Software Unit Testing - (IEEE/ANSI Standard 1008),
* IEEE Standard for Software Quality Assurance Plans - (IEEE/ANSI Standard 730) etc.

Other software development/IT management process assessment methods besides CMMI and ISO 9000 include SPICE, Trillium, TickIT, Bootstrap, ITIL, MOF, and CobiT.

**2.7 Software Process**

Process: “A particular method of doing some thing, generally involving a number of steps or operations is a process. In software engineering, the phrase software process refers to the method of developing software.

The process that deals with the technical and management issues of software development is called **Software Process**. Clearly, many different types of activities need to be performed to develop software. A software development project must have atleast development activities and project management activities. All these activities together comprise software process.

**2.7.1 Processes, Projects, Products**

* A software process specifies a method of developing software.
* A software project, on the other hand, is a development project in which a software process is used.
* A software product is the outcome of a software project.

Project k

**Software Process**

Project 1

Project j

…..

…..

Product2

Product 1

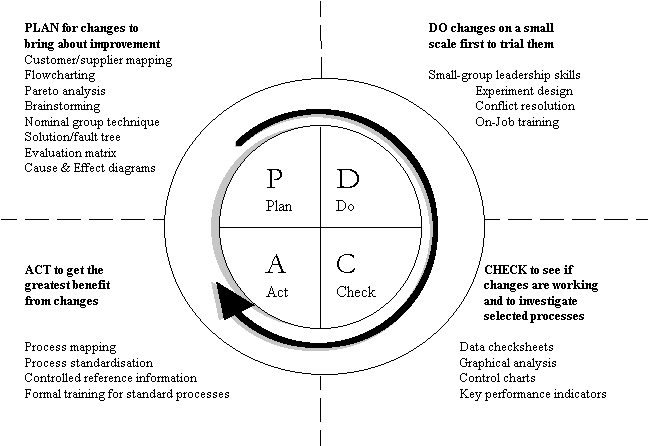
Product m

**Fig 2.2 Software Process Workflow**

Each software development project starts with some needs and hopefully ends with some software that satisfies those needs. A software process specifies the abstract the abstract set of activities that should be performed to go from user needs to final product.

**2.8 Shewhart’s PDCA or Deming’s Wheel**

The concept of the PDCA (Plan-Do-Check-Act) Cycle, often also and quite rightly referred to as `the Shewhart Cycle' was originally developed by Walter Shewhart, a statistician in the Bell Laboratories who developed statistical process during the 1930's

****The Continuous Improvement Cycle

**Fig 2.3 PDCA with example tools for every step**

Testing is a key component of software development processes. Like all process the software development process follows a cycle comprised of the following four components:

* **Plan (P):** Device a plan. Define your objective and determine the strategy and support methods required to achieve that objective.
* **Do (D):** Execute the plan. Create the conditions and perform the necessary training to execute the plan.
* **Check (C):** Check to determine whether work is progressing according to the plan and whether the results are obtained.
* **Action (A):** Take the necessary action. If your checkup reveals that the work is not being performed according to plan or that results are not what anticipated device measures for appropriate action was.
  1. **Summary**
* Quality of software can be expressed by a collection of parameters that may include adequacies of fulfillment of all requirement specifications- functional, behavioral, performance and interface etc., reliability, robustness and fault/failure tolerance, maintainability (good documentation and process standards play a vital role here), user-friendliness, usability, etc.
* Quality is obviously a subjective term that entirely depends on relative judgments and interpretations from different viewpoints of the stakeholders involved e.g. the developer, the tester, the auditor, the customer and the market in general.
* The concept of the PDCA (Plan-Do-Check-Act) Cycle, often also and quite rightly referred to as `the Shewhart Cycle' was originally developed by Walter Shewhart, a statistician in the Bell Laboratories who developed statistical process during the 1930's.
* According to ISO 9000 definitions, Quality Control is the operational techniques and activities that are used to fulfill requirements for quality and Quality Assurance includes all the planned and systematic activities implemented to provide adequate confidence that an entity will fulfill requirements for quality.
* Also common are project teams that include a mix of testers and developers who work closely together, with overall QA processes monitored by project managers. It primarily depends on what best fits an organization's size and business structure.
* Software quality systems consist of:
* SEI-CMM = (process-oriented, organizational-level applications)
* ISO = (International Organization for Standardization) - The ISO 9001:2000 standard
* IEEE = (Institute of Electrical and Electronics Engineers) standards
* Other software development/IT management process assessment methods besides CMMI and ISO 9000 include SPICE, Trillium, TickIT, Bootstrap, ITIL, MOF, and CobiT.

**Review Questions**

1. What is the uniqueness of software quality in the context of the debate between software seen as a process vs. as a product?
2. Can you really measure all parameters of software quality?
3. Why is quality in software a subjective matter?
4. What roles do the auditors play in quality certifications?
5. What are the specific application domains for:
   1. CMM
   2. CMMI
   3. PCMM
   4. ISO
6. Why is there a need for different types of process quality standards?
7. Why is there no specific software product quality standard?