

UNIT VIII:

Syllabus: *Quality Management: Quality concepts, Software quality assurance, Software Reviews, Formal technical reviews, Statistical Software quality Assurance, Software reliability, And the ISO 9000 quality standards.*

1. Write the notes about quality management and quality concept?

Some software developers continue to believe that software quality is something you begin to worry about after code has been generated. Nothing could be further from the truth! *Software quality assurance* (SQA) is an umbrella activity that is applied throughout the software process

Software quality is important, you have to (1) explicitly define what is meant when you say “software quality,” (2) create a set of activities that will help ensure that every software engineering work product exhibits high quality, (3) perform quality assurance activities on every software project, (4) use metrics to develop strategies for improving your software process and, as a consequence, the quality of the end product.

Who does it? Everyone involved in the software engineering process is responsible for quality. Why is it important? You can do it right, or you can do it over again. If a software team stresses quality in all software engineering activities, it reduces the amount of rework that it must do. That results in lower costs, and more importantly, improved time-to-market.

What are the steps? Before software quality assurance activities can be initiated, it is important to define ‘software quality’ at a number of different levels of abstraction. Once you understand what quality is, a software team must identify a set of SQA activities that will filter errors out of work products before they are passed on.

What is the work product? A Software Quality Assurance Plan is created to define a software team’s SQA strategy. During analysis, design, and code generation, the primary SQA work product is the formal technical review summary report. During testing, test plans and procedures are produced. Other work products associated with process improvement may also be generated.

SQA encompasses (1) a quality management approach, (2) effective software engineering technology (methods and tools), (3) formal technical reviews that are applied throughout the software process, (4) a multitiered testing strategy, (5) control of software documentation and the changes made to it, (6) a procedure to ensure compliance with software development standards (when applicable), and (7) measurement and reporting mechanisms.

QUALITY CONCEPTS

It has been said that no two snowflakes are alike. Certainly when we watch snow falling it is hard to imagine that snowflakes differ at all, let alone that each flake possesses a unique structure. In order to observe differences between snowflakes, we must examine the specimens closely, perhaps using a magnifying glass. In fact, the closer we look the

more differences we are able to observe. This phenomenon, *variation between samples*, applies to all products of human as well as natural creation. For example, if two “identical” circuit boards are examined closely enough, we may observe that the copper pathways on the boards differ slightly in geometry, placement, and thickness. In addition, the location and diameter of the holes drilled in the boards varies as well. All engineered and manufactured parts exhibit variation. The variation between samples may not be obvious without the aid of precise equipment to measure the geometry, electrical characteristics, or other attributes of the parts. However, with sufficiently sensitive instruments, we will likely come to the conclusion that no two samples of any item are exactly alike.

Variation control is the heart of quality control. A manufacturer wants to minimize the variation among the products that are produced, even when doing something relatively simple like duplicating diskettes. Surely, this cannot be a problem—duplicating diskettes is a trivial manufacturing operation, and we can guarantee that exact duplicates of the software are always created.

Quality

The *American Heritage Dictionary* defines *quality* as “a characteristic or attribute of something.” As an attribute of an item, quality refers to measurable characteristics—things we are able to compare to known standards such as length, color, electrical properties, and malleability. However, software, largely an intellectual entity, is more challenging to characterize than physical objects.

Quality of design refers to the characteristics that designers specify for an item. The grade of materials, tolerances, and performance specifications all contribute to the quality of design. As higher-grade materials are used, tighter tolerances and greater levels of performance are specified, the design quality of a product increases, if the product is manufactured according to specifications.

Quality of conformance is the degree to which the design specifications are followed during manufacturing. Again, the greater the degree of conformance, the higher is the level of quality of conformance. In software development, quality of design encompasses requirements, specifications, and the design of the system. Quality of conformance is an issue focused primarily on implementation. If the implementation follows the design and the resulting system meets its requirements and performance goals, conformance quality is high. But are quality of design and quality of conformance the only issues that software engineers must consider? Robert Glass argues that a more “intuitive” relationship is in order:

User satisfaction = compliant product + good quality + delivery within budget and schedule

At the bottom line, Glass contends that quality is important, but if the user isn’t satisfied, nothing else really matters. DeMarco reinforces this view when he states: “A product’s quality is a function of how much it changes the world for the better.” This view of quality contends that if a software product provides substantial benefit to its end-users, they may be willing to tolerate occasional reliability or performance problems.

Quality Control: *Quality control* involves the series of inspections, reviews, and tests used throughout the software process to ensure each work product meets the requirements placed upon it. Quality control includes a feedback loop to the process that created the work product. The combination of measurement and feedback allows us to tune the process when the work products created fail to meet their specifications. This approach views quality control as part of the manufacturing process. Quality control activities may be fully automated, entirely manual, or a combination of automated tools and human interaction. A key concept of quality control is that all work products have defined, measurable specifications to which we may compare the output of each process. The feedback loop is essential to minimize the defects produced.

Quality Assurance: *Quality assurance* consists of the auditing and reporting functions of management. The goal of quality assurance is to provide management with the data necessary to be informed about product quality, thereby gaining insight and confidence that product quality is meeting its goals. Of course, if the data provided through quality assurance identify problems, it is management's responsibility to address the problems and apply the necessary resources to resolve quality issues.

Cost of Quality: The *cost of quality* includes all costs incurred in the pursuit of quality or in performing quality-related activities. Cost of quality studies are conducted to provide a base-line for the current cost of quality, identify opportunities for reducing the cost of quality, and provide a normalized basis of comparison. The basis of normalization is almost always dollars. Once we have normalized quality costs on a dollar basis, we have the necessary data to evaluate where the opportunities lie to improve our processes. Furthermore, we can evaluate the effect of changes in dollar-based terms.

Quality costs may be divided into costs associated with prevention, appraisal, and failure. *Prevention costs* include

- quality planning
- formal technical reviews
- test equipment
- training

Appraisal costs include activities to gain insight into product condition the "first time through" each process. Examples of appraisal costs include

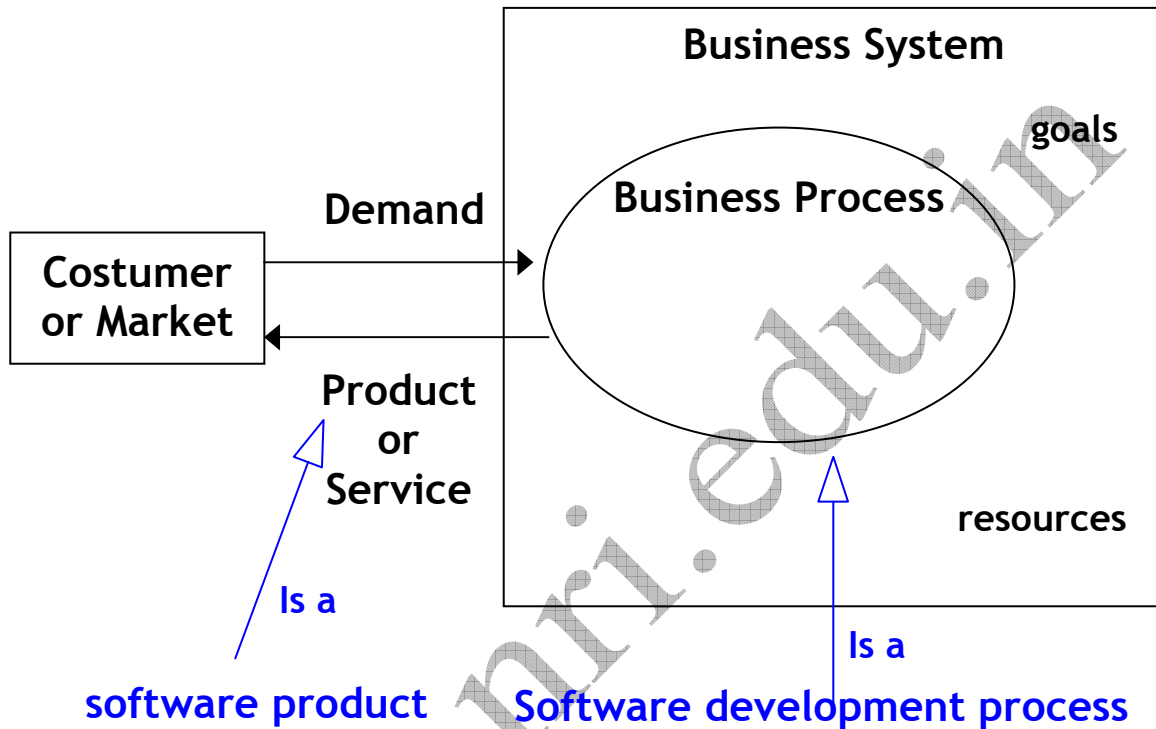
- in-process and interprocess inspection
- equipment calibration and maintenance
- testing

2. The objectives of software quality

1. To introduce the notion of software quality and describe common software quality attributes and quality-factors
2. To introduce the verification and validation process
3. To introduce the software quality management process and key quality management activities
4. To explain the role of standards in software quality management

5. To explain the main approaches to verification and validation and quality control: testing, inspections and reviews and measurements
6. To explain why formal development methods are important to improve software quality

Product and process:



Quality of service:

Some product-related services and their quality attributes

- User Training
- User Help
 - Quick and useful response (avoid “Help does not Help”)
- Product repair and new versions deployment
 - Quick and effective repair
 - Conservation qualities:
 - Things that worked well in the old version, continue to work well in the new version (regression tests are very important here), and don’t require new user training
 - Installation of the new version doesn’t cause loss of user data (backward compatibility)
 - Installation of the new version doesn’t require system down for too much time
 - Progress qualities:

- Things that worked wrong or didn't work at all in the old version, now work well in the new version, or new useful features have been added

Quality-related activities

Software Verification and Validation (V & V)

- Goals:
 - Establish the existence of defects in a product
 - Assess whether or not the product is usable in an operational situation
- Verification
 - Ensure that we are **building the product right**, i.e., according to its specification
- Validation
 - Ensure that we are **building the right product**, i.e., according to user needs
- V & V are integral part of the development process
- Concerned directly with product quality

Software Quality Management (SQM)

- Goals: Ensure that the required level of quality is achieved in software products, namely, that defined standards and procedures are followed
- SQM should aim to develop a 'quality culture' where quality is seen as everyone's responsibility
- Sub-activities:
 - (Organization-wide) **Quality assurance**
 - Establish organisational procedures and standards for quality in a quality manual
 - (Project-wide) **Quality planning**
 - Select applicable procedures and standards for a particular project and modify these as required. Produce a quality plan.
 - (Project-wide) **Quality control (QC)**
 - Ensure that procedures and standards are followed by the software development team. Produce quality review reports
- Quality management should be separate from project management to ensure independence of budget and schedule pressures
- Concerned directly with process quality and, indirectly, product quality

Main approaches for V&V and QC

Tests

- Dynamic technique, concerned with exercising and observing product behaviour to discover **defects**
- The system is executed with test data (defined test cases) and its operational behaviour is observed to discover defects (differences between observed and expected)

- Used mainly for V & V
- GUI testing difficult to automate; API testing easier to automate

Inspections and reviews

- Static technique - concerned with the analysis of the static system representation (source code, documentation, ...) to discover **problems**
- May be supplemented by tool-based document and code analysis

Measurements

- The value of defined metrics is automatically measured on selected components of the product, for prediction or control purposes
- Used mainly for CQ

All involve planning, execution and result analysis and reporting

Quality assurance and standards

1. Standards are the key to effective quality management
2. They may be international, national, organizational or project standards
3. Product standards define characteristics that all components should exhibit e.g. a common programming style
4. Process standards define how the software process should be enacted

Importance of standards:

1. Encapsulation of best practice - avoids repetition of past mistakes
2. Framework for quality assurance process – it involves checking standard compliance
3. Provide continuity - new staff can understand the organisation by understand the standards applied

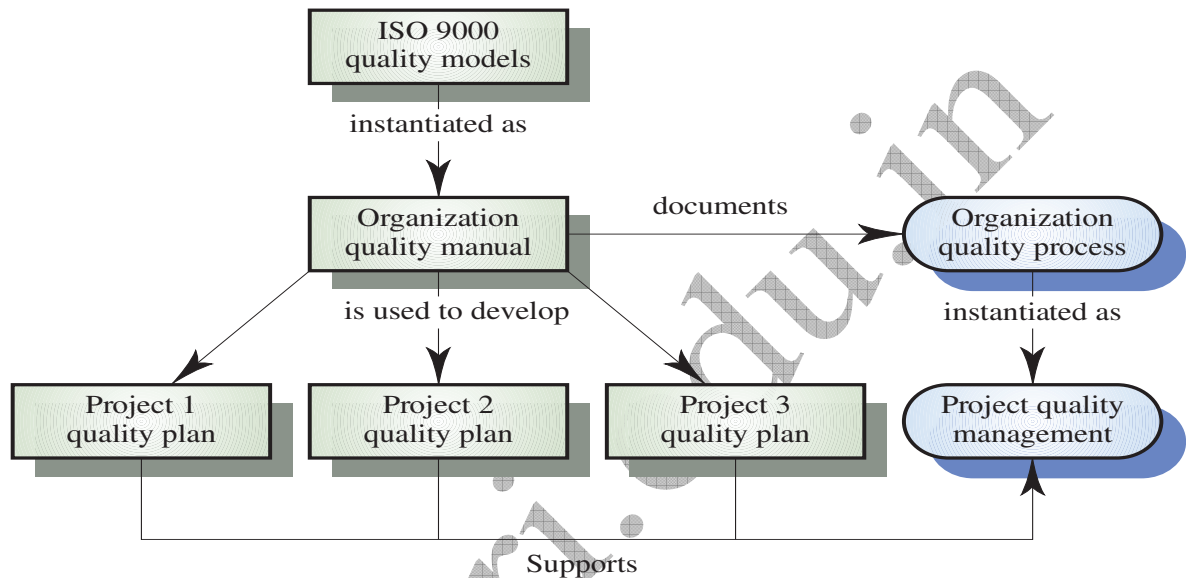
4. Enumerate the ISO 9000 software quality standards in detail?

1. International set of standards for quality management (ISO 9000:2000, ISO 9001:2000, ISO 9004:2000, etc.)
2. Applicable to a range of organisations from manufacturing to service industries
3. ISO 9001:2000 specifies requirements for a quality management system for any organization that needs to demonstrate its ability to consistently provide product that meets customer and applicable regulatory requirements and aims to enhance customer satisfaction, in all business sectors
 - a. Integrates previous standards ISO 9001, ISO 9002 and ISO 9003
 - b. ISO 9001 is a generic model that must be instantiated for each organisation
4. ISO 9004:2000 provides guidance for continual improvement of a quality management system to benefit all parties (employees, owners, suppliers, society in general...) through sustained customer satisfaction. It should be used to extend the benefits obtained from ISO 9001:2000 to all parties that are interested in or affected by the business operations.

ISO 9000 certification

- ❖ Quality standards and procedures should be documented in an organisational quality manual.
- ❖ External body may certify that an organisation’s quality manual conforms to ISO 9000 standards (namely ISO 9001)
- ❖ Customers are, increasingly, demanding that suppliers are ISO 9000 certified

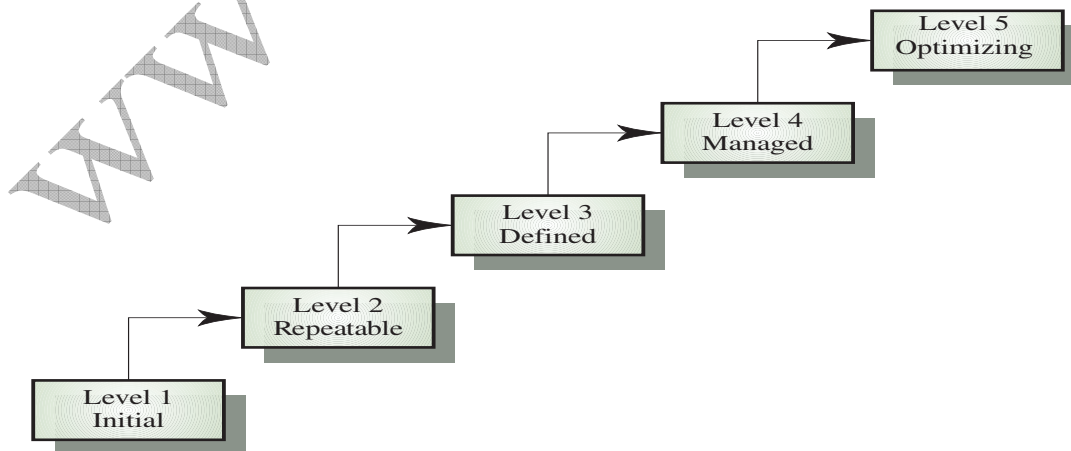
ISO 9000 and quality management



The Software Engineering Institute (SEI) Capability Maturity Model for Software (CMM)

Is a model for?

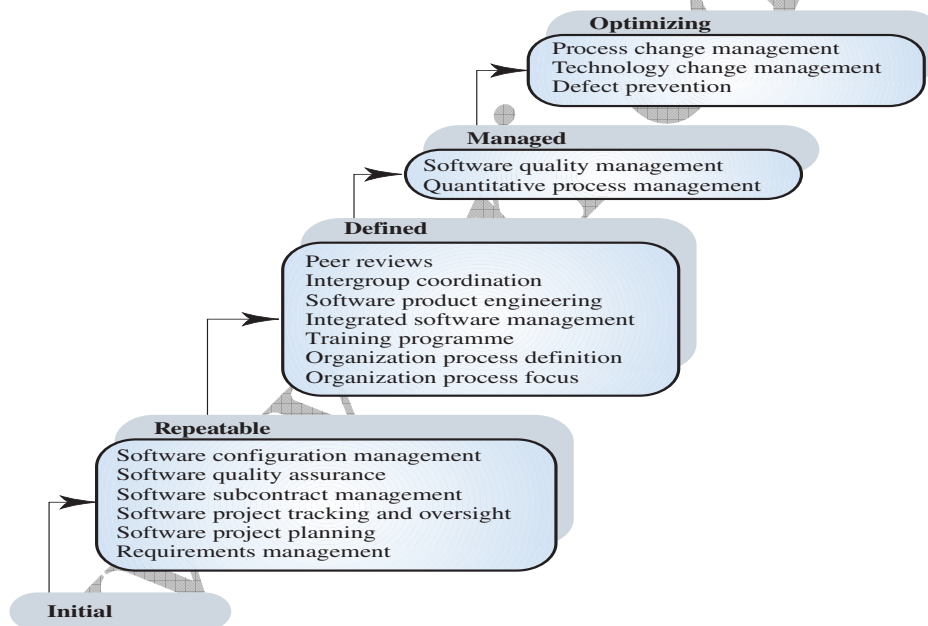
- ✓ judging the maturity of the software processes of an organization
- ✓ identifying the key practices that are required to increase the maturity of these processes



CMM maturity levels

- 1) **Initial.** The software process is characterized as ad hoc, and occasionally even chaotic. Few processes are defined, and success depends on individual effort and heroics.
- 2) **Repeatable.** Basic project management processes are established to track cost, schedule, and functionality. The necessary process discipline is in place to repeat earlier successes on projects with similar applications.
- 3) **Defined.** The software process for both management and engineering activities is documented, standardized, and integrated into a standard software process for the organization. All projects use an approved, tailored version of the organization's standard software process for developing and maintaining software.
- 4) **Managed.** Detailed measures of the software process and product quality are collected. Both the software process and products are quantitatively understood and controlled.
- 5) **Optimizing.** Continuous process improvement is enabled by quantitative feedback from the process and from piloting innovative ideas and technologies.

CMM key process areas



The CMM and ISO 9000

- There is a clear correlation between the key processes in the CMM and the quality management processes in ISO 9000
- The CMM is more detailed and prescriptive and includes a more detailed framework for improvement
- Organisations rated as level 2 in the CMM are likely to be ISO 9000 compliant

5. Illustrate the Software Quality Assurance in software engineering?

1 Software Quality

Definition: *Software quality is called the conformance to explicitly stated functional and Performance equirements, documented development standards, and implicit characteristics.*

Important points:

- software requirements are the foundation from which quality is measured ;
- specified standards define development criteria that guide the manner in which the software is engineered ;
- if the software meets only the explicit requirements, and does not meet the implicit requirements, the software quality is suspect.

2 Software Quality factors

Operational characteristics:

- correctness - does it do what I want?
- reliability - does it do it accurately?
- efficiency - will it run efficiently on my hardware?
- integrity - is it secure?
- usability - is it designed for the user?

Product revision:

- maintainability - can I fix it?
- flexibility - can I change it?
- testability - can I test it?

Product transition:

- portability - will I be able to use it on another machine?
- reusability - will I be able to reuse some of the software?
- interoperability - will I be able to interface it with another system?

6. What are the Metrics for Grading the Software Quality factors?

- auditability - the ease with which conformance to standards can be checked
- accuracy - the precision of computations and control
- communication commonality - the degree to which standard interfaces are used

- completeness - the degree to which the implementation has been achieved
- conciseness - the compactness of the program in terms of lines of code
- consistency - the use of uniform design and documentation techniques
- data commonality - the use of standard data structures and types
- error tolerance - the damage that occurs when the program encounters an error
- execution efficiency - the run-time performance of the program
- expandability - the degree to which the design can be extended
- generality - the breadth of potential application of program components
- hardware independence - the degree of decoupling from the hardware
- modularity - the functional independence of program components
- operability - the ease of operation with the system
- security - existence of mechanisms that protect the data and the program
- simplicity - the degree of understandability of the program without difficulty
- traceability - the ability to trace a component back to the requirements

7. Discuss about the various steps in Software Quality System?

The quality factors are developed in a system called a quality system, or quality management system.

The software quality system consists of the managerial structure, responsibilities, activities, capabilities and resources to ensure that the developed software products have the desired quality.

The quality management system encompasses the following activities:

- reviews of the projects' qualities
- career development of staff
- development of standards and procedures

The concrete details of the quality management system will be contained in a quality manual. *A quality manual will contain standards, procedures and guidelines and will be influenced by external standards.*

- a standard is instruction of how a project document or program code is to be displayed ;

- a procedure is a step-by-step set of instructions describing how a particular software activity is to be carried out ;

- a guideline - consists of advice on best practice.

2. Software Reviews

Software reviews are a filter to the software engineering process. Reviews are applied at various points during software development and serve to uncover defects that can be removed.

A software review is a way of using a group of people to:

- point out needed improvements in the product of a single person or team
- confirm those parts of the product in which improvement is not desired
- achieve technical work of more uniform quality than can be achieved without reviews, in order to make technical work more manageable

There are many different types of reviews:

- informal meetings
- Formal presentation of software
- Walkthroughs

The obvious benefit from formal technical reviews, walkthroughs, is the early discovery of software defects so that each defect may be corrected prior to the next step in the software engineering process.

A defect amplification model can be used to illustrate the generation and detection of errors during the steps in the software engineering process.

3. Formal Technical Reviews

A formal technical review (FTR) is a software quality assurance activity

Performed by software engineers with the following objectives:

- to uncover errors in function, logic or implementation of the software ;
- to verify that the software meets its requirements ;
- to ensure that the software has been developed according to the standards ;
- to achieve uniform software ;
- to make projects manageable.

The formal technical review serves to promote backup and continuity because a number of people become familiar with parts of the software that they may not have otherwise seen.

Each FTR is conducted as a meeting and is considered successful only if it is properly planned, controlled and attended.

The Review Meeting

Every review meeting should abide by the following constraints:

- between 3 and 5 people should be involved in the review ;
- advance preparation should occur but should require no more than 2 hours of work for each person;
- the duration of the review meeting should be less than 2 hours.

Rather than attempting to review the entire design, walkthroughs are conducted for modules or for small groups of modules.

The focus of the FTR is on a product- a component of the software.

The review meeting is attended by the review leader, all reviewers, and the producer.

The producer organizes a "walk through" the product, explaining the material, while the reviewers raise issues based on their advance preparation.

When errors are discovered, the recorder notes each.

At the end of the review the attendees decide whether to accept the product or not, with or without modifications.

4. Software Quality Standards

The most general standards are: ISO 9001 and British Standard Institute BS5750

The relevant standards for the industry are:

ISO 9001: Quality Systems-Model for Quality Assurance and Design

ISO 9000-3: Guidelines for the application of ISO 9001 to the Development,

Supply and Maintenance of Software

ISO 9004-2: Quality Management and Quality System Elements