SE-3002
SOFTWARE QUALITY ENGINEERING

RUBAB JAFFAR

RUBAB.JAFFAR@NU.EDU.PK

Quality types & quality cost

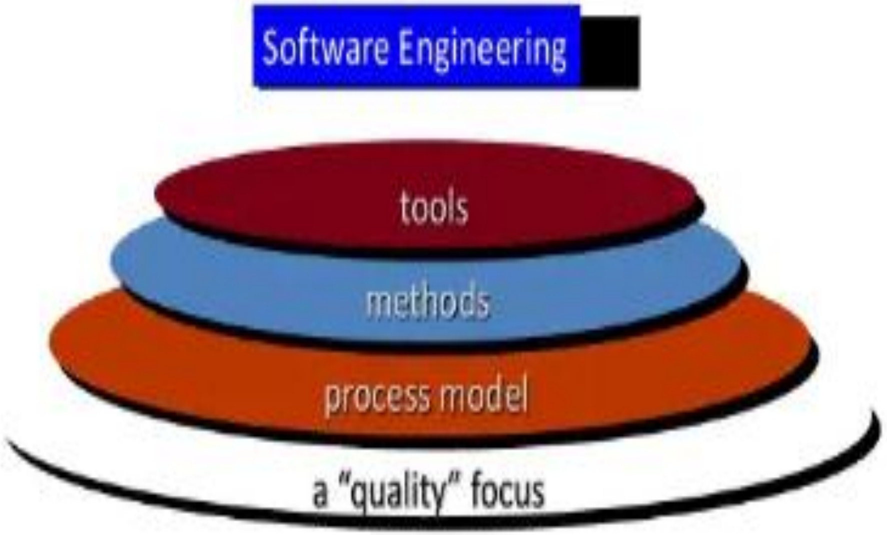
Overview and Basics

Lecture # 4,5,6
13,14,16 Sep

TODAY’S OUTLINE

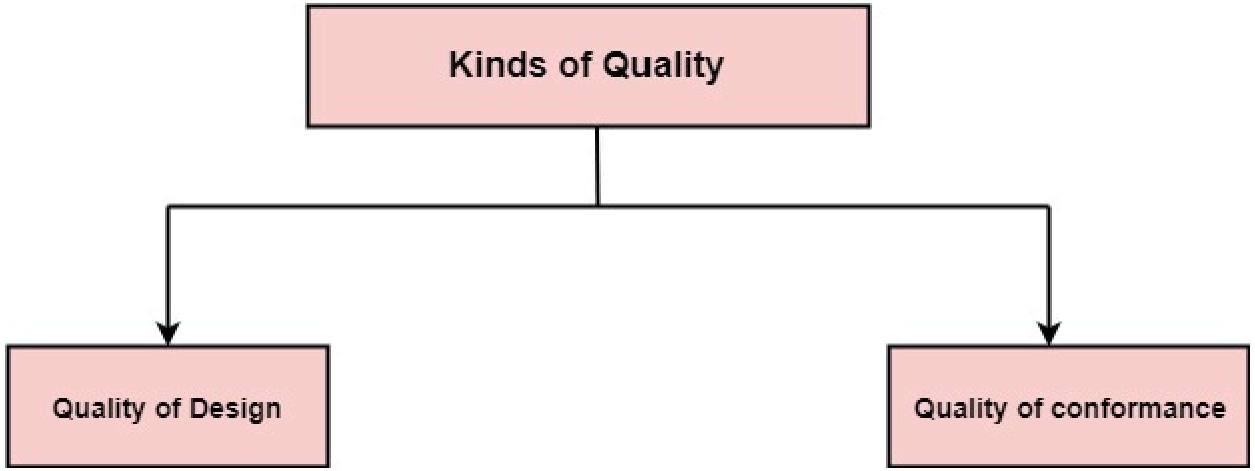
- Types of Quality
- QA VS QC
- Cost of Quality
- SQE activities
- SQE activities and software processes
- QIP
- Effort profile

QUALITY


Software Engineering
tools
methods
process model
a "quality" focus

KIND OF QUALITY

- Quality?
- Two kinds of quality
  - Quality of design
  - Quality of Conformance



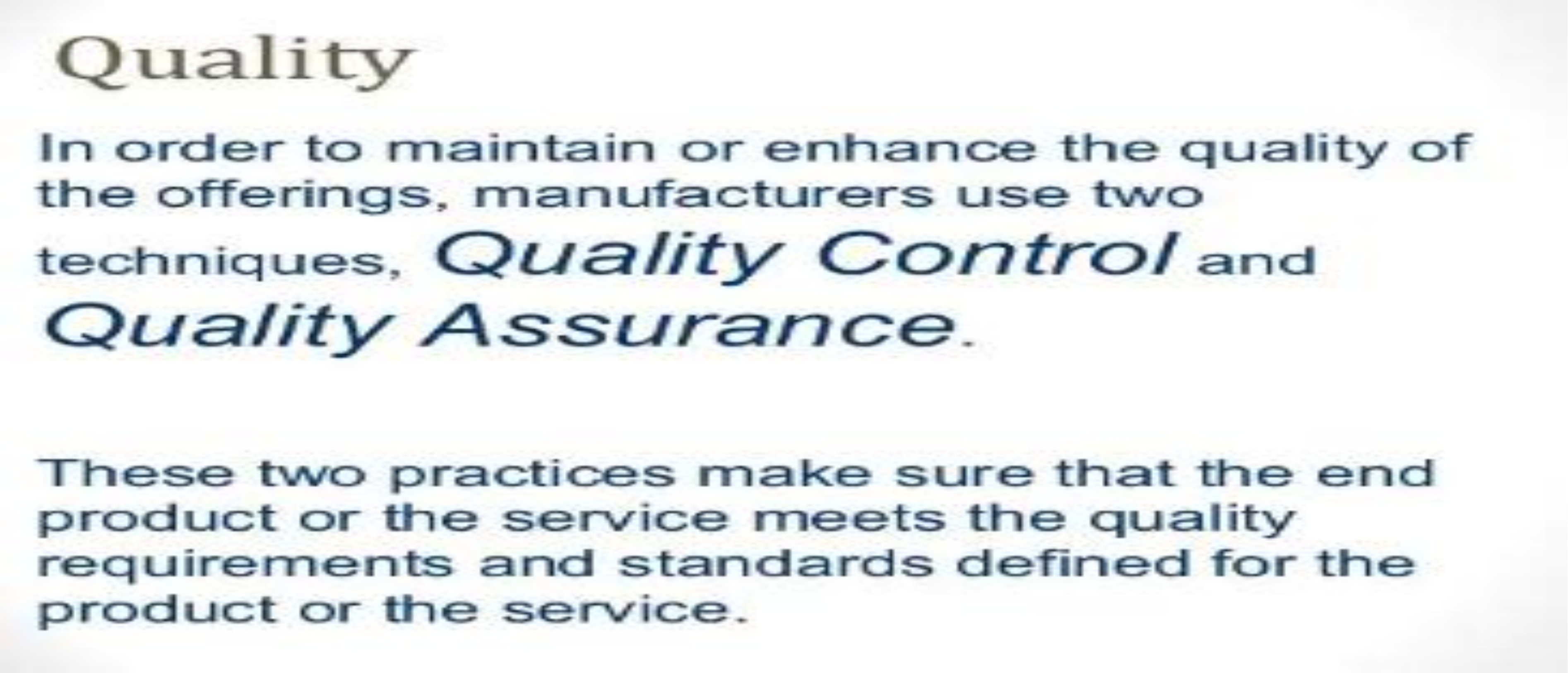
QUALITY OF DESIGN

- Quality of design is the degree to which the designers is intending to provide its customers.
- In the manufacturing industry, an example of quality of design could be the difference between a Honda and a Mercedes. Both cars provide implied uses ,however both cars differ in their features and are designed for two different sets of customers.
- In the service industry, an example of quality of design could be the difference between an economy hotel and a five star hotel. Both provide basic need for customers ,however their different features are designed for two different customers.

QUALITY OF CONFORMANCE

- Quality of conformance is the degree to how well a product or service meets its specifications.
- In the manufacturing industry, an example of quality conformance could be a wrench that has a specific dimension for it's intended use that can be observed and measured.
- In the service industry, an example of quality of conformance could be the implied wait for food in the restaurant to be fifteen to twenty minutes. When the food takes thirty minutes to arrive then it can be assumed that there has been poor quality of conformance.

QC AND QA



Quality

In order to maintain or enhance the quality of the offerings, manufacturers use two techniques, *Quality Control* and *Quality Assurance*.

These two practices make sure that the end product or the service meets the quality requirements and standards defined for the product or the service.

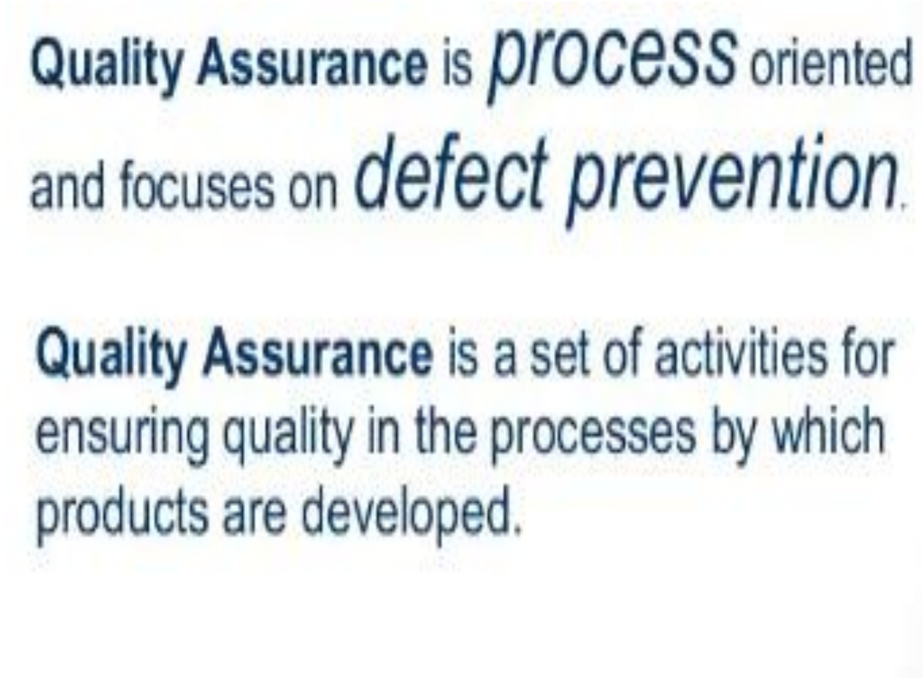
ARE QA AND QC SAME TERMS?

- BIG NO, these both terms are effectively different.
- Most of the time we use both terms randomly, hence to study and understand the difference between them is important.

ARE QA AND QC SAME TERMS?

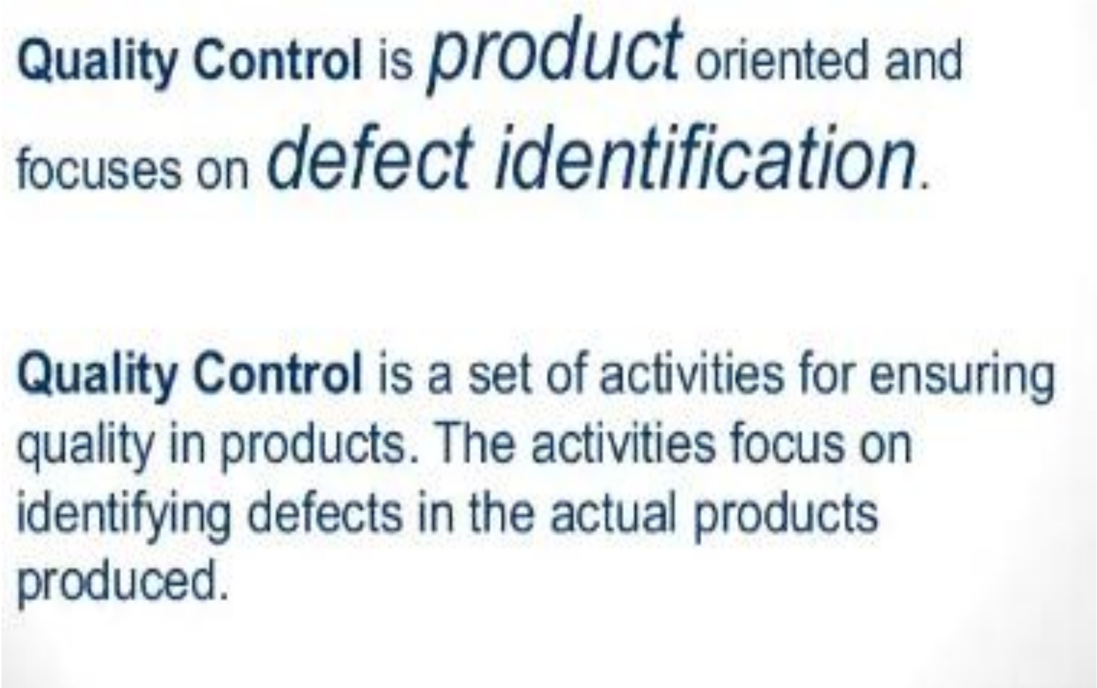
- Lets differentiate according to the below points.
- Definition
- Focus on
- Goal
- How to achieve
- Example
- Responsibility

DEFINITION OF QUALITY ASSURANCE AND QUALITY CONTROL



Quality Assurance is *process* oriented and focuses on *defect prevention*.

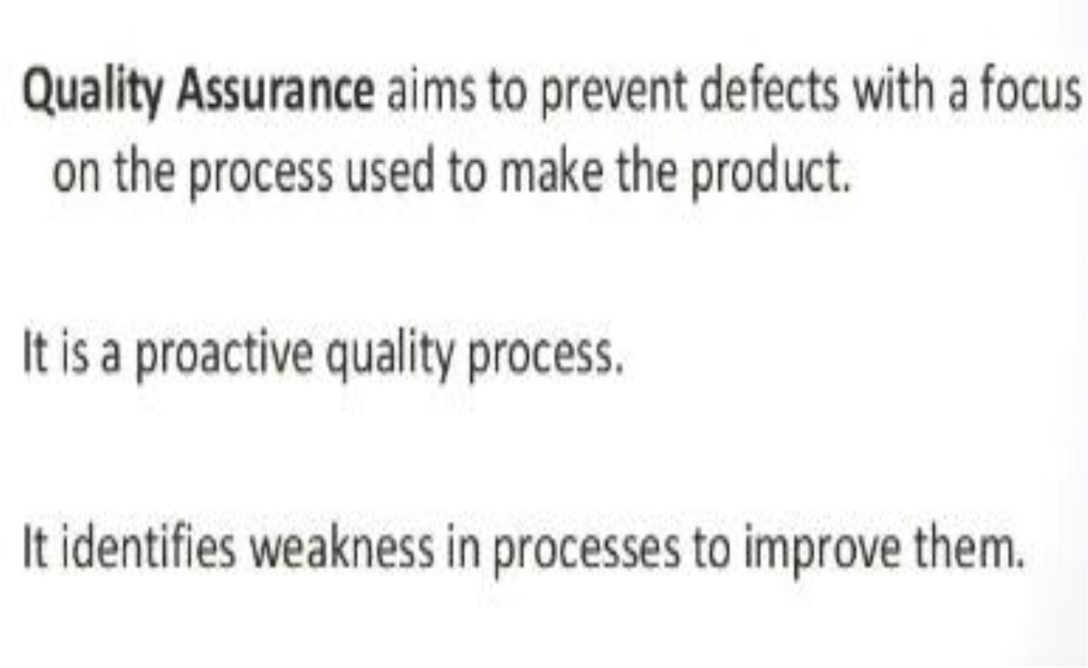
Quality Assurance is a set of activities for ensuring quality in the processes by which products are developed.



Quality Control is *product* oriented and focuses on *defect identification*.

Quality Control is a set of activities for ensuring quality in products. The activities focus on identifying defects in the actual products produced.

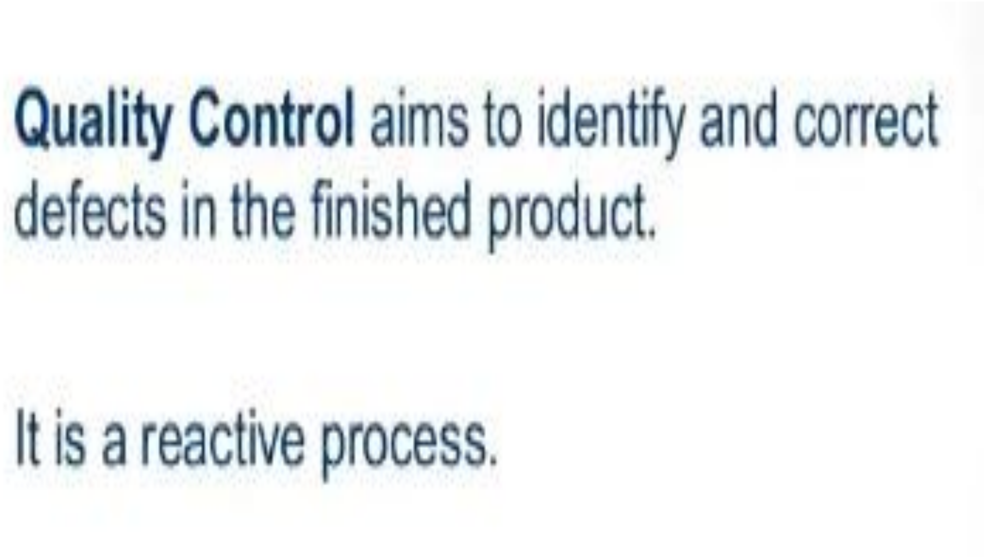
FOCUSES ON



Quality Assurance aims to prevent defects with a focus on the process used to make the product.

It is a proactive quality process.

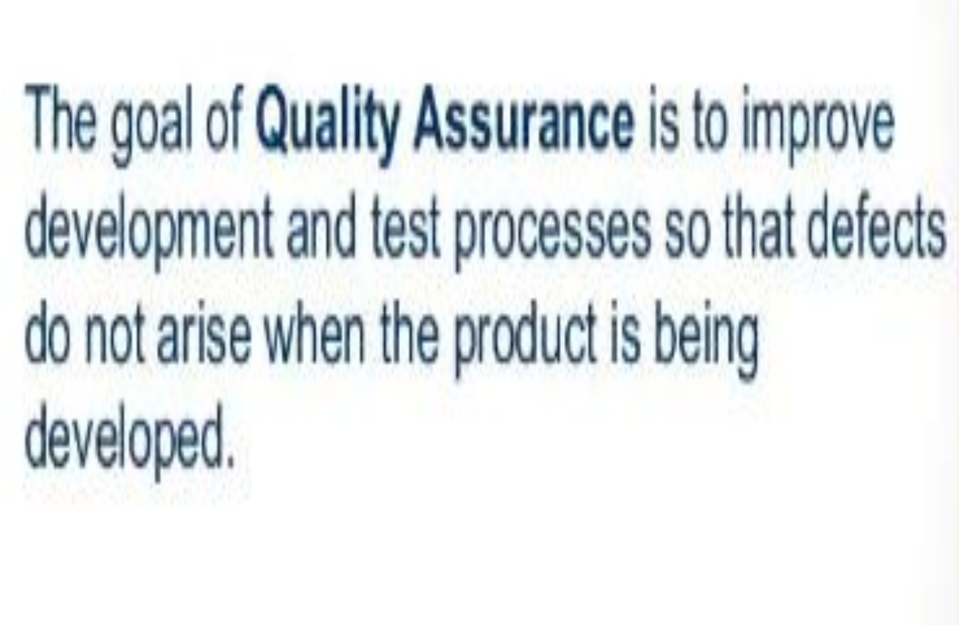
It identifies weakness in processes to improve them.

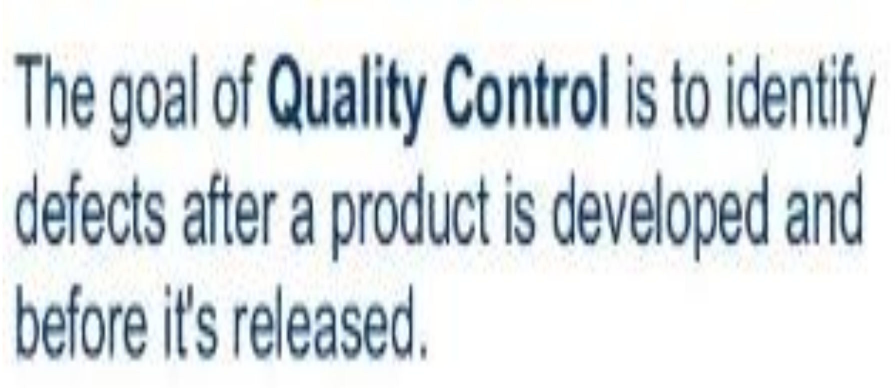


Quality Control aims to identify and correct defects in the finished product.

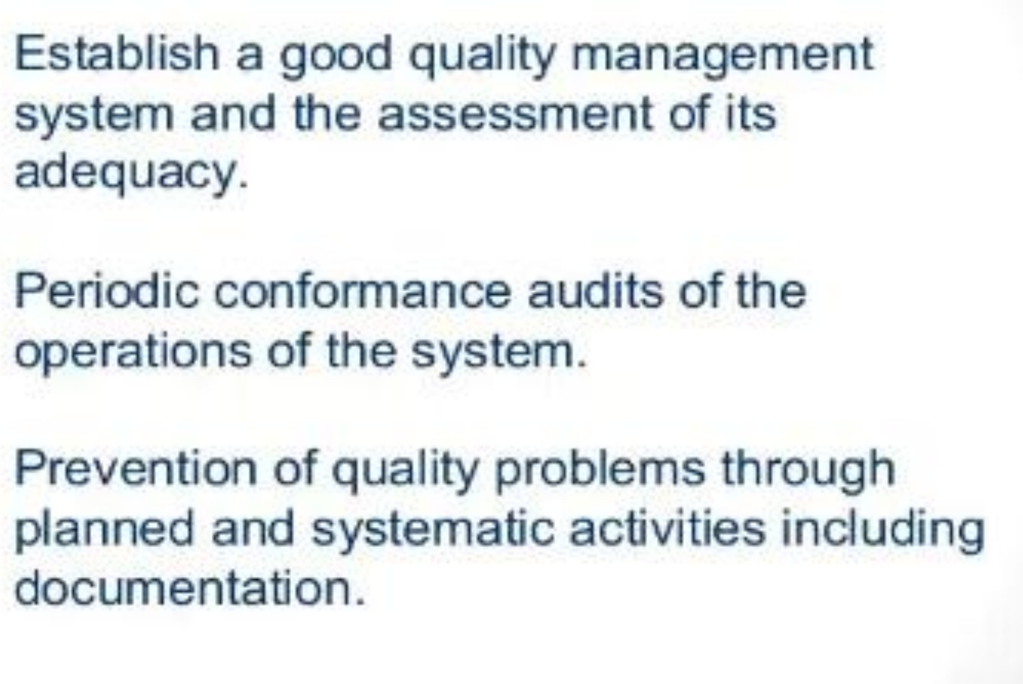
It is a reactive process.

GOAL OF QA


The goal of **Quality Assurance** is to improve development and test processes so that defects do not arise when the product is being developed.


The goal of **Quality Control** is to identify defects after a product is developed and before it's released.

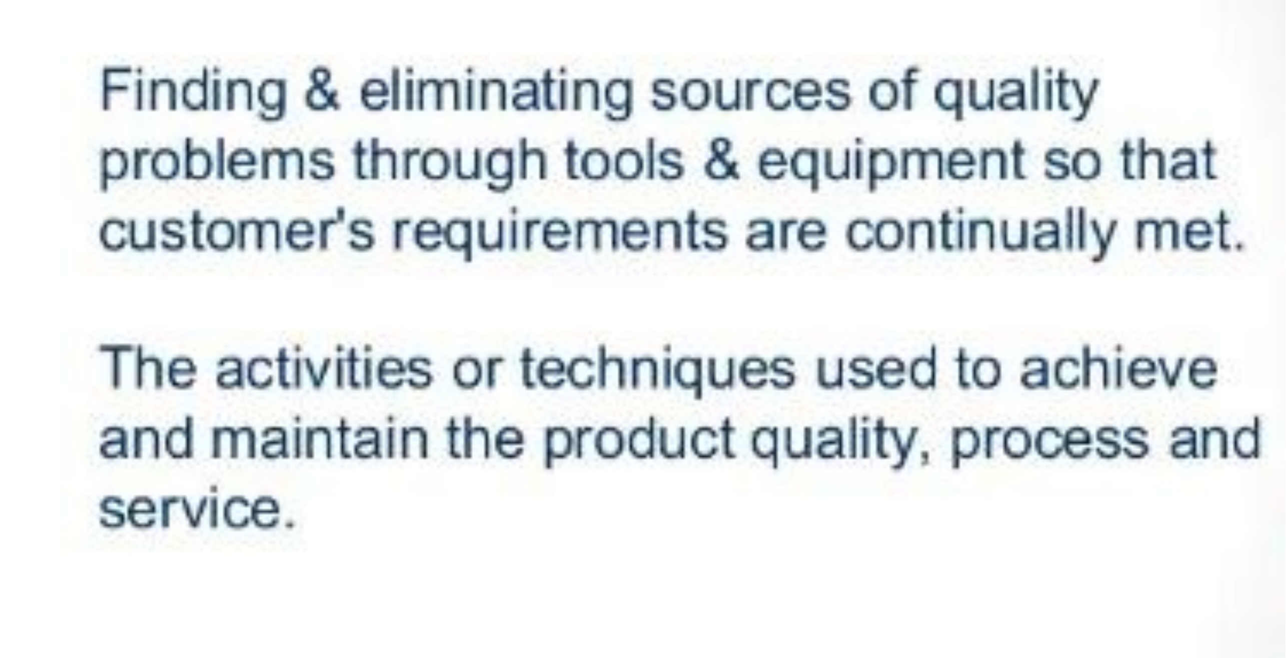
TO ACHIEVE GOALS



Establish a good quality management system and the assessment of its adequacy.

Periodic conformance audits of the operations of the system.

Prevention of quality problems through planned and systematic activities including documentation.

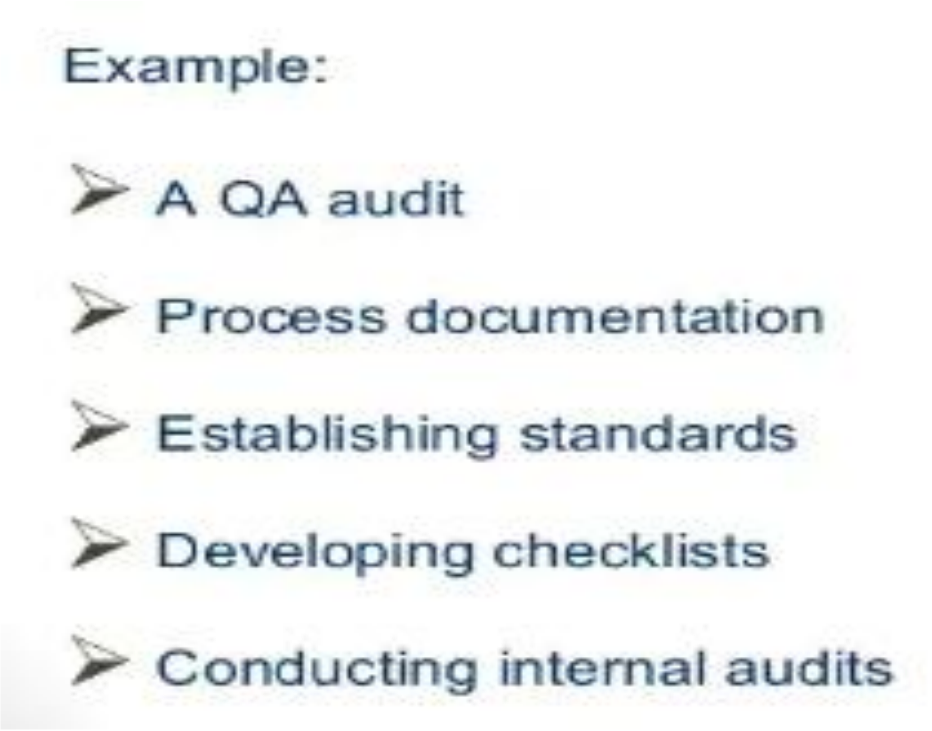


Finding & eliminating sources of quality problems through tools & equipment so that customer's requirements are continually met.

The activities or techniques used to achieve and maintain the product quality, process and service.

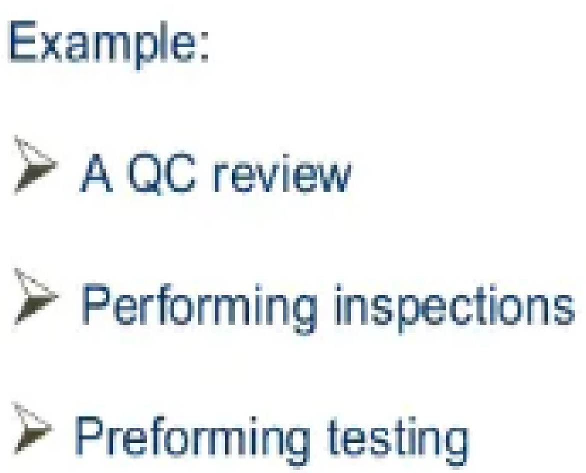
QA & QC EXAMPLES

Example



Example:

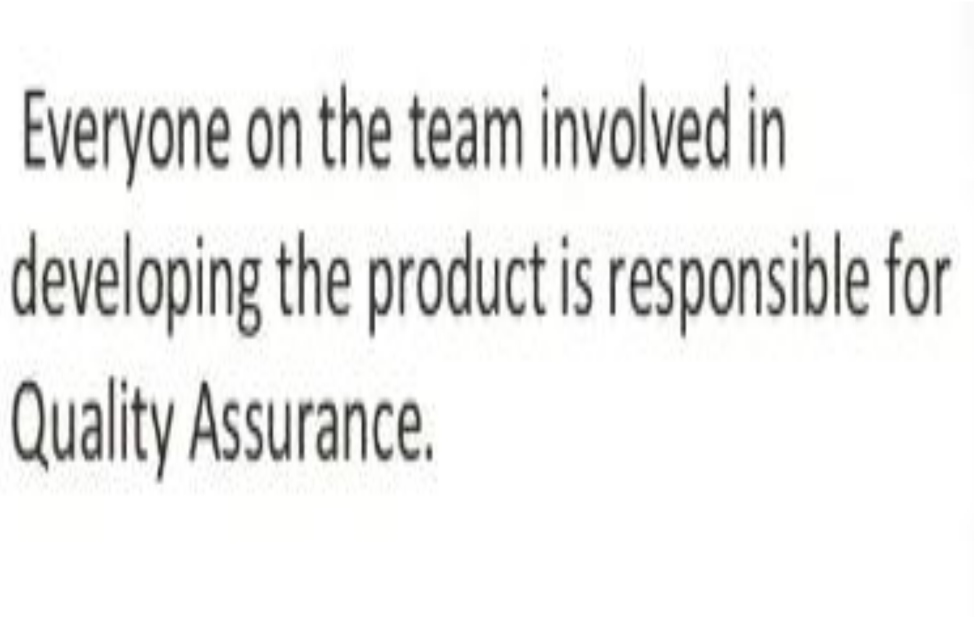
- A QA audit
- Process documentation
- Establishing standards
- Developing checklists
- Conducting internal audits

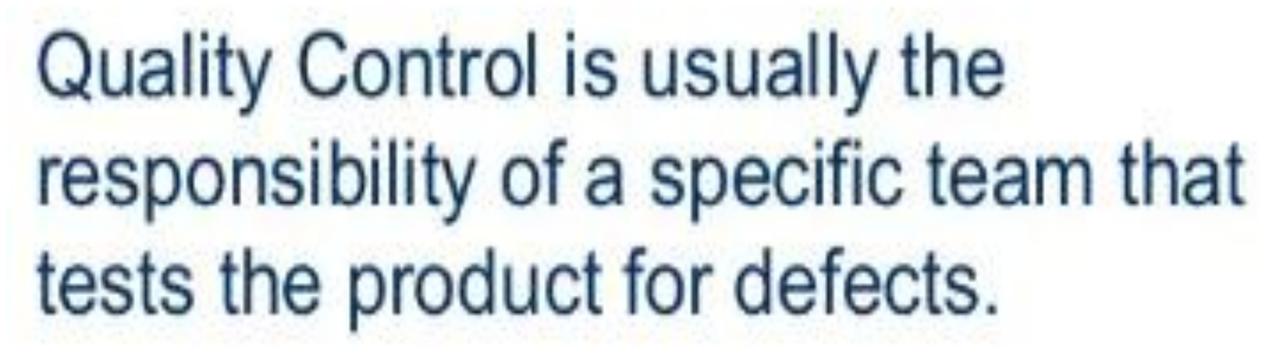


Example:

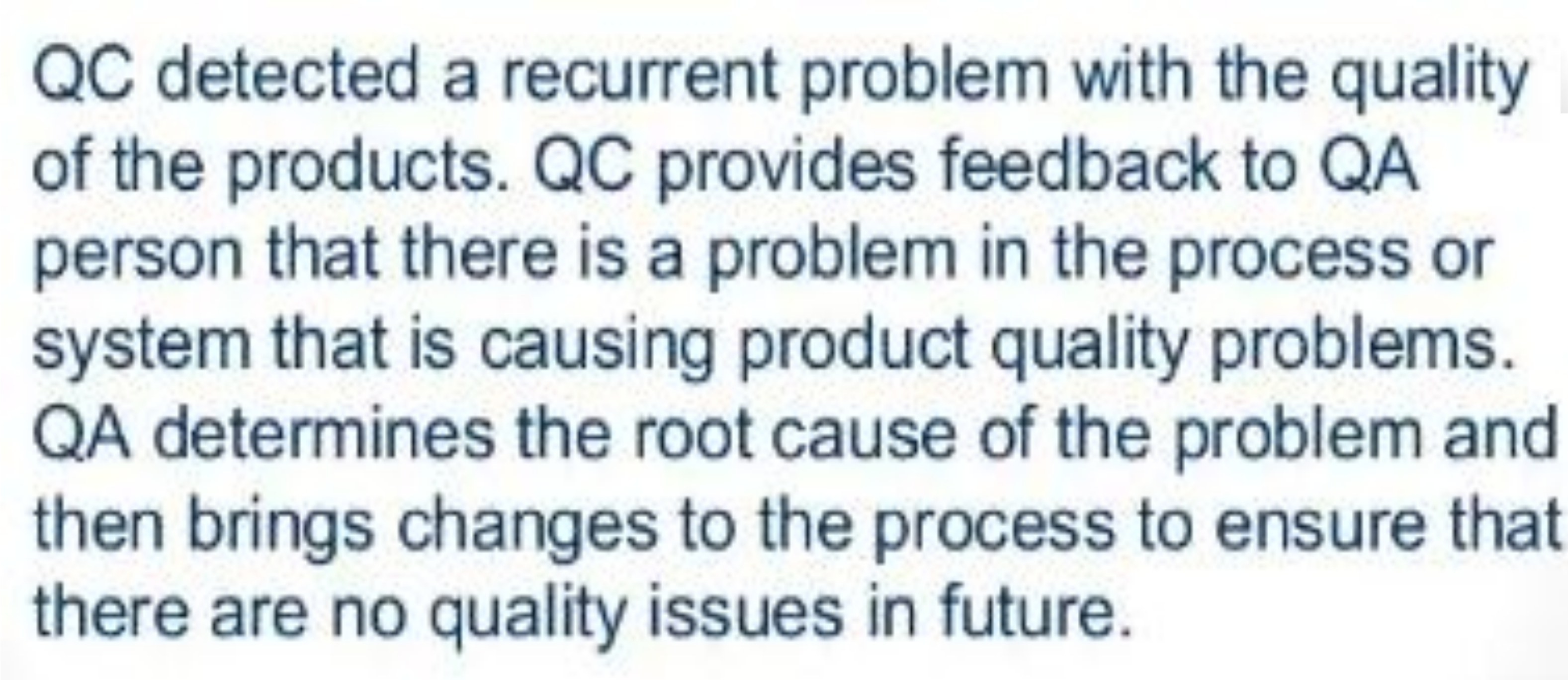
- A QC review
- Performing inspections
- Preforming testing

RESPONSIBLE FOR QA & QC


Everyone on the team involved in developing the product is responsible for Quality Assurance.


Quality Control is usually the responsibility of a specific team that tests the product for defects.

QA AND QC


QC detected a recurrent problem with the quality of the products. QC provides feedback to QA person that there is a problem in the process or system that is causing product quality problems. QA determines the root cause of the problem and then brings changes to the process to ensure that there are no quality issues in future.

DEFECTS IN THE CONTEXT OF QA AND QUALITY ENGINEERING

- Three generic ways to deal with defects include:
  - defect prevention
  - defect detection and removal
  - defect containment
- Quality engineering can also be viewed as defect management. In addition to the execution of the planned QA activities, quality engineering also includes:
  - quality planning before specific QA activities are carried out
  - measurement, analysis, and feedback to monitor and control the QA activities

CAUSES OF SOFTWARE ERRORS

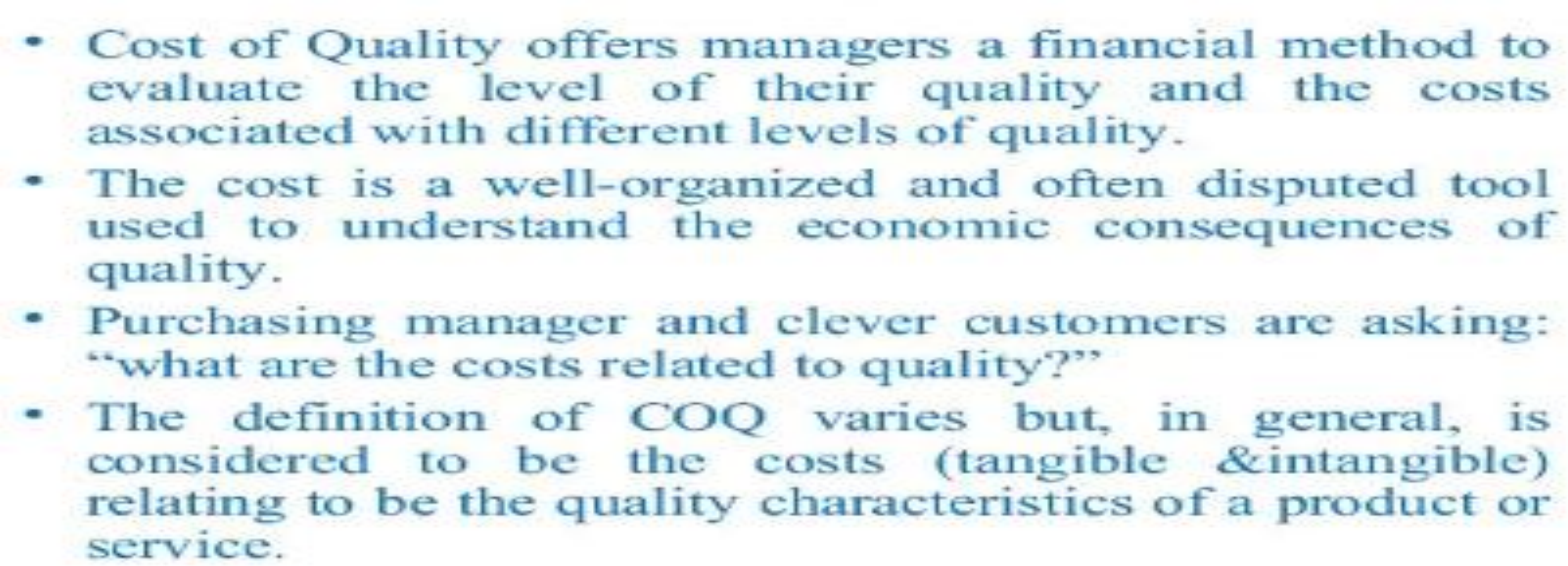
- Faulty definition of requirements
  - Erroneous definition of requirements
  - Incomplete definition of requirements
  - Inclusion of unnecessary requirements, functions that are not expected to be needed in the near future
- Client-developer communication failures
  - Misunderstanding of (i) client's instructions as stated in the requirement documents, (ii) requirement changes, (iii) client's response to design problem
- Deliberate deviation from requirements
- Logical design errors
  - Process definitions that contain sequencing errors
  - Erroneous definition of boundary conditions
  - Omission of required software states
  - Omission of definitions concerning reactions to illegal operation of the software system.

CAUSES OF SOFTWARE ERRORS

- Coding errors
  - misunderstanding the design documentation, linguistic errors in the programming languages, errors in the application of CASE and other development tools, errors in data selection, and so forth.
- Non-compliance with documentation and coding instructions
  - Team members who need to coordinate their own codes with code modules developed by “non-complying” team members can be expected to encounter more than the usual number of difficulties when trying to understand the software developed by the other team members.
  - Individuals replacing the “non-complying” team member (who has retired or been promoted) will find it difficult to fully understand his or her work.
  - The design review team will find it more difficult to review a design prepared by a non-complying team.
- Shortcomings of the testing process
  - Incomplete test plans; failure to report, document and correct detected errors;
- Procedure errors
- Documentation errors
- Omission of software functions

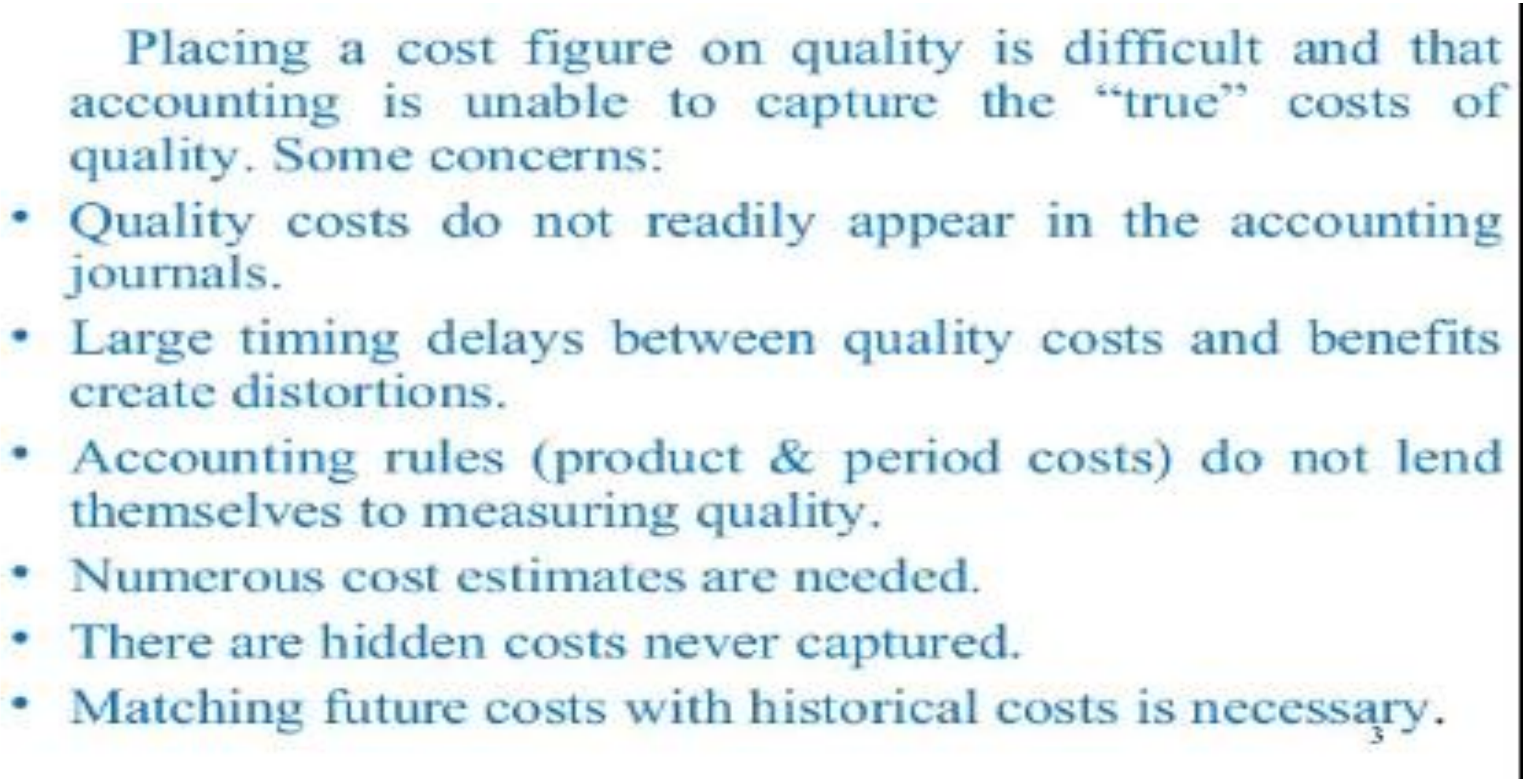
- COST of Quality(CoQ)

DEFINITION



- Cost of Quality offers managers a financial method to evaluate the level of their quality and the costs associated with different levels of quality.
- The cost is a well-organized and often disputed tool used to understand the economic consequences of quality.
- Purchasing manager and clever customers are asking: “what are the costs related to quality?”
- The definition of COQ varies but, in general, is considered to be the costs (tangible &intangible) relating to be the quality characteristics of a product or service.

DIFFICULTIES IN CAPTURING THE "TRUE COST"



Placing a cost figure on quality is difficult and that accounting is unable to capture the "true" costs of quality. Some concerns:

- Quality costs do not readily appear in the accounting journals.
- Large timing delays between quality costs and benefits create distortions.
- Accounting rules (product & period costs) do not lend themselves to measuring quality.
- Numerous cost estimates are needed.
- There are hidden costs never captured.
- Matching future costs with historical costs is necessary.

COST OF SOFTWARE QUALITY (COSQ)

- *Cost of software quality* – the economic assessment of software quality development and maintenance – is just another class of software quality metrics, where financial values are used as the measuring tool.
- Application of common financial measures enables management to obtain the type of general overview of all software quality assurance activities unavailable with any other metrics.
- Application of cost of software quality metrics enables management to achieve economic control over SQA activities and outcomes. The specific objectives are:
  - Control of organization-initiated costs to prevent and detect software errors
  - Evaluation of the economic damages of software failures as a basis for revising the SQA budget
  - Evaluation of plans to increase or decrease SQA activities or to invest in a new or updated SQA infrastructure on the basis of past economic performance

OBJECTIVES OF COST OF SOFTWARE QUALITY METRICS

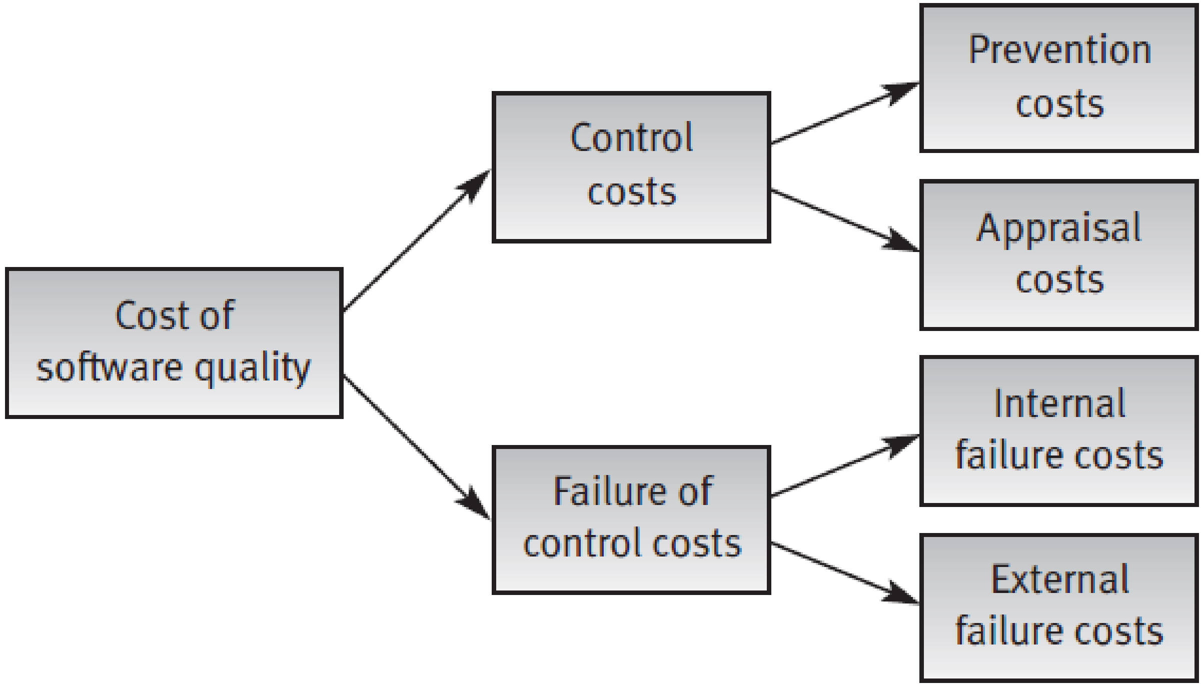
- Managerial control over the cost of software quality is achieved by comparison of actual performance figures with:
  - Control budgeted expenditures (for SQA prevention and appraisal activities)
  - Previous year's failure costs
  - Previous project's quality costs (control costs and failure costs)
  - Other department's quality costs (control costs and failure costs).

COST METRICS/PARAMETERS FOR THE EVALUATION OF AN SQA PLAN

- Percentage of cost of software quality out of total software development costs
- Percentage of software failure costs out of total software development costs
- Percentage of cost of software quality out of total software maintenance costs
- Percentage of cost of software quality out of total sales of software products and software maintenance.

THE CLASSIC MODEL OF COST OF SOFTWARE QUALITY

- The classic quality cost model, developed in the early 1950s by Feigenbaum and others, provides a methodology for classifying the costs associated with product quality assurance from an economic point of view.
- The model classifies costs related to product quality into two general classes:



THE CLASSIC MODEL OF COST OF SOFTWARE QUALITY

- **Costs of control** include costs that are spent to prevent and detect software errors in order to reduce them to an accepted level.
- **Costs of failure of control** include costs of failures that occurred because of failure to prevent and detect software errors.
- **Prevention costs** include investments in quality infrastructure and quality activities that are not directed to a specific project or system, being general to the organization.
- **Appraisal costs** include the costs of activities performed for a specific project or software system for the purpose of detecting software errors.

THE CLASSIC MODEL OF COST OF SOFTWARE QUALITY

- **Internal failure costs** include costs of correcting errors that have been detected by design reviews, software tests and acceptance tests (carried out by the customer) and completed before the software is installed at customer sites.
- **External failure costs** include all costs of correcting failures detected by customers or the maintenance team after the software system has been installed

THE CLASSIC MODEL OF COST OF SOFTWARE QUALITY

- **Prevention costs**
  - Investments in development of new or improved SQA infrastructure components or, alternatively, regular updating of those components:
    - Procedures and work instructions
    - Support devices: templates, checklists, etc.
    - Software configuration management system
    - Software quality metrics (product, process, project)
  - Regular implementation of SQA preventive activities:
    - Instruction of new employees in SQA subjects and procedures related to their positions
    - Instruction of employees in new and updated SQA subjects and procedures
    - Certification of employees for positions that require special certification
    - Consultations on SQA issues provided to team leaders and others.

THE CLASSIC MODEL OF COST OF SOFTWARE QUALITY

- **Prevention costs**
  - Control of the SQA system through performance of:
    - Internal quality reviews
    - External quality audits by customers and SQA system certification organizations
    - Management quality reviews.

THE CLASSIC MODEL OF COST OF SOFTWARE QUALITY

- **Appraisal costs**
  - Reviews:
    - Formal design reviews (DRs)
    - Peer reviews (inspections and walkthroughs)
    - Expert reviews
  - Costs of software testing
    - Unit tests
    - Integration tests
    - Software system tests
    - Acceptance tests (participation in tests carried out by the customer).
  - Costs of assuring quality of external participants, primarily by means of design reviews and software testing. These activities are applied to the activities performed by:
    - Subcontractors
    - Suppliers of COTS software systems and reusable software modules
    - The customer as a participant in performing the project.

THE CLASSIC MODEL OF COST OF SOFTWARE QUALITY

- **Internal failure costs**
  - internal failure costs represent the costs of error correction subsequent to formal examinations of the software during its development, prior to the system's installation at the customer's site.
    - Costs of redesign or design corrections subsequent to design review and test findings
    - Costs of re-programming or correcting programs in response to test findings
    - Costs of repeated design review and re-testing (regression tests).
    - Importantly, although the costs of regular design reviews and software tests are considered appraisal costs, any repeated design reviews or software tests directly resulting from poor design and inferior code quality are considered internal failure costs

THE CLASSIC MODEL OF COST OF SOFTWARE QUALITY

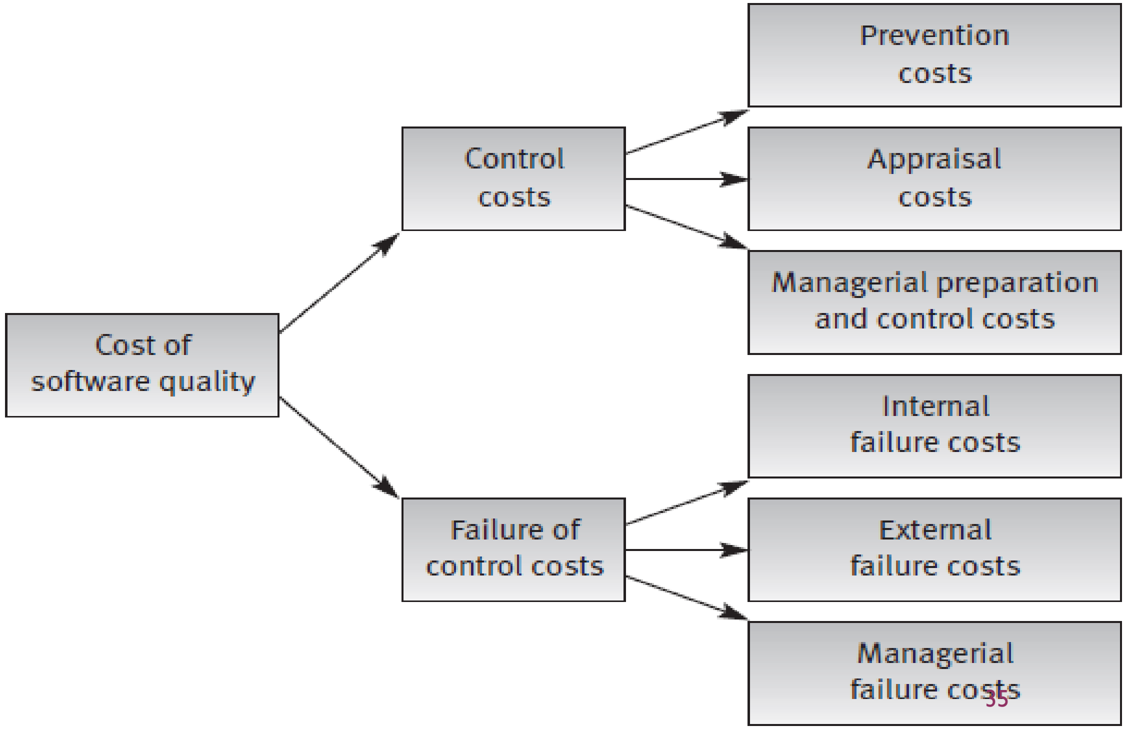
- **External failure costs**
  - External failure costs entail the costs of correcting failures detected by customers or maintenance teams after the software system has been installed at customer sites.
  - These costs may be further classified into "overt" external failure costs and "hidden" external failure costs. In most cases, the extent of hidden costs is much greater than that of overt costs. Typical **overt external failure** costs include:
    - Resolution of customer complaints during the warranty period
    - Correction of software bugs detected during regular operation (often performed at customer's site)
    - Correction of software failures after the warranty period is over even if the correction is not covered by the warranty.
    - Damages paid to customers in case of a severe software failure detected during regular operation.
    - Reimbursement of customer's purchase costs, including handling, in case of total dissatisfaction (relates to COTS software packages as well as to custom-made software).
    - Insurance against customer's claims in case of severe software failure

THE CLASSIC MODEL OF COST OF SOFTWARE QUALITY

- Typical examples of **hidden external failure** costs are:
  - Damages of reduction of sales to customers suffering from high rates of software failures
  - Severe reduction of sales motivated by the firm's damaged reputation
  - Increased investment in sales promotion to counter the effects of past software failures
  - Reduced prospects to win a tender or, alternatively, the need to underprice to prevent competitors from winning tenders.

AN EXTENDED MODEL FOR COST OF SOFTWARE QUALITY

- The *extended cost of software quality model*, extends the classic model to include management's "contributions" to the total cost of software quality. According to the extended model, two subclasses are added to complete the model's coverage: managerial preparation and control costs, and managerial failure costs.



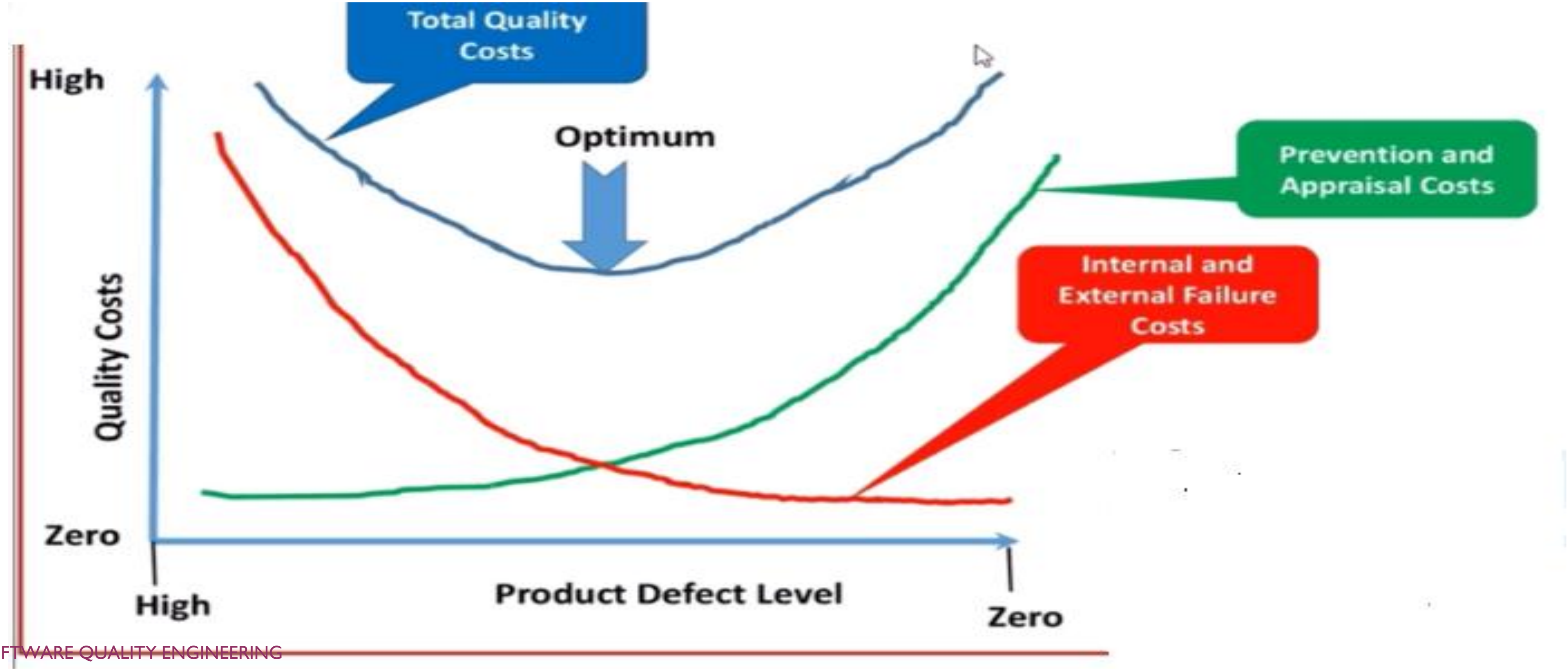
AN EXTENDED MODEL FOR COST OF SOFTWARE QUALITY

- **Managerial preparation and control costs**
  - Costs of carrying out contract reviews (proposal draft and contract draft reviews).
  - Costs of preparing project plans, including quality plans and their review.
  - Costs of periodic updating of project and quality plans.
  - Costs of performing regular progress control of internal software development efforts.
  - Costs of performing regular progress control of external participants' contributions to the project.

AN EXTENDED MODEL FOR COST OF SOFTWARE QUALITY

- **Managerial failure costs**
  - Managerial failure costs can be incurred throughout the entire course of software development, beginning in the pre-project stage. Typical managerial failure costs include:
    - Unplanned costs for professional and other resources, resulting from underestimation of the resources upon which the submitted proposals are based.
    - Damages paid to customers as compensation for late completion of the project, a result of the unrealistic schedule presented in the company's proposal.
    - Damages paid to customers as compensation for late completion of the project, a result of management's failure to recruit sufficient and appropriate team members.
    - *Domino effect*: damages to other projects performed by the same teams involved in the delayed projects. These damages should be considered managerial failure costs of the original project, whose scheduling problems interfered with the progress of other projects.

RELATION BETWEEN DIFFERENT QUALITY COST



QUALITY ENGINEERING

- Different customers and users have different quality expectations under different market environments. Therefore, we need to move beyond just performing QA activities toward quality engineering by managing these quality expectations as an engineering problem.
  - A goal might be to minimize the cost and project risk

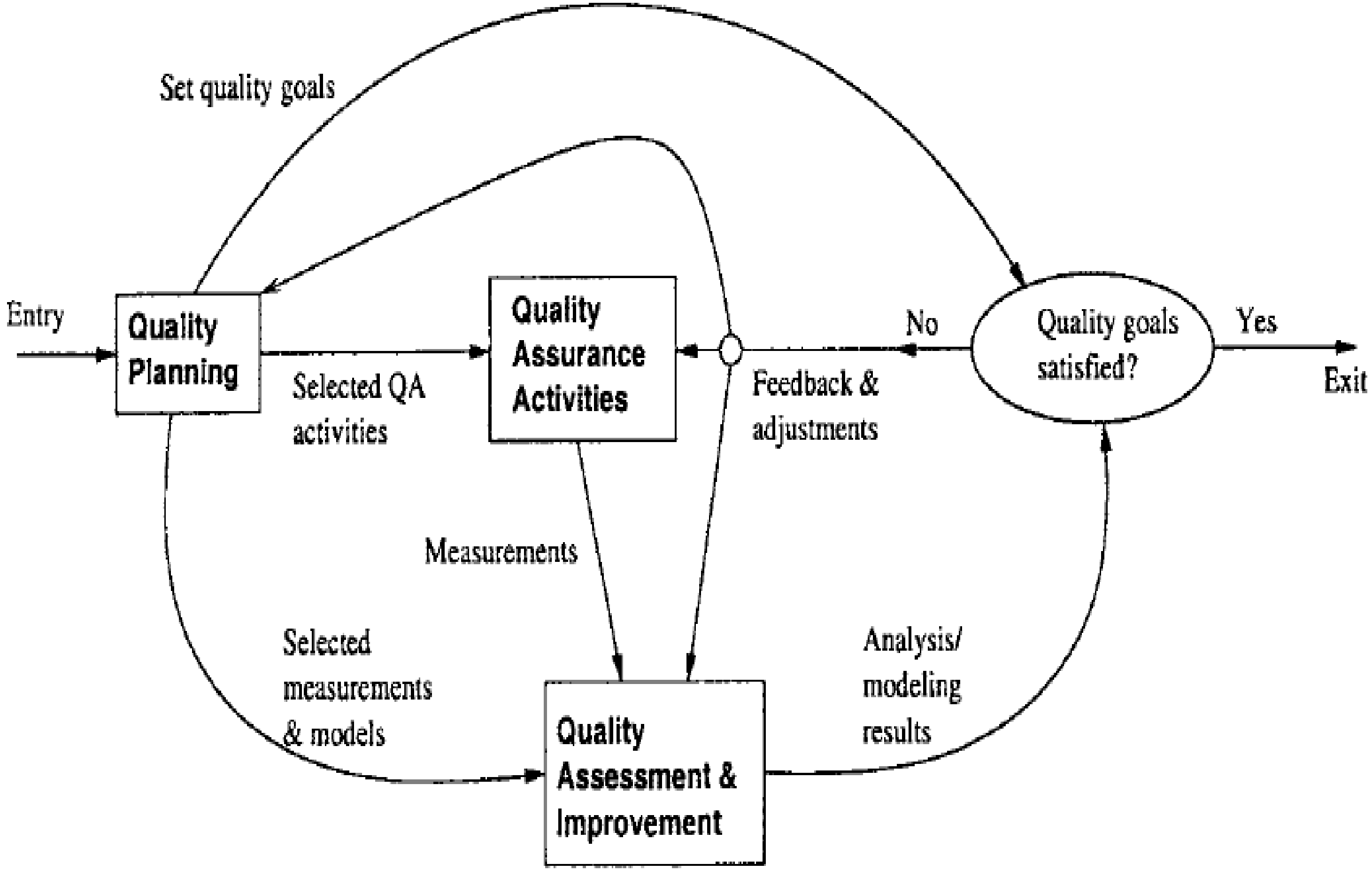
SOFTWARE QUALITY ENGINEERING

1) The application of a continuous, systematic, disciplined, quantifiable approach to the development and maintenance of quality throughout the whole life cycle of software products and systems; that is, the application of quality engineering to software,
2) The study of approaches as in (1).

SQE- COOPERATION

QUALITY ENGINEERING: ACTIVITIES AND PROCESS

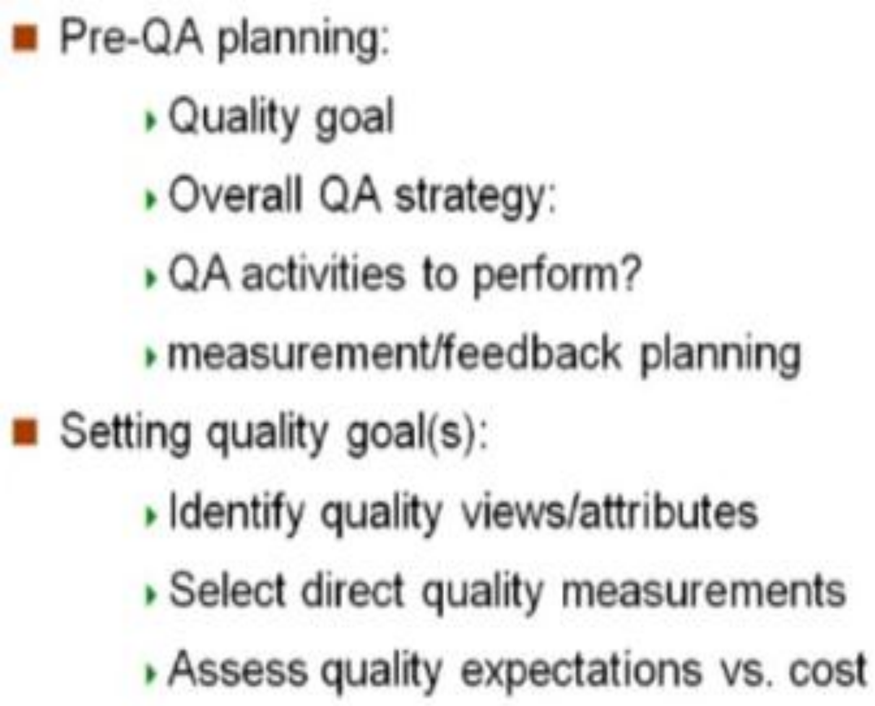
- There are three major groups of activities in the quality engineering process:
  - pre-QA activities, in-QA activities, and post-QA activities



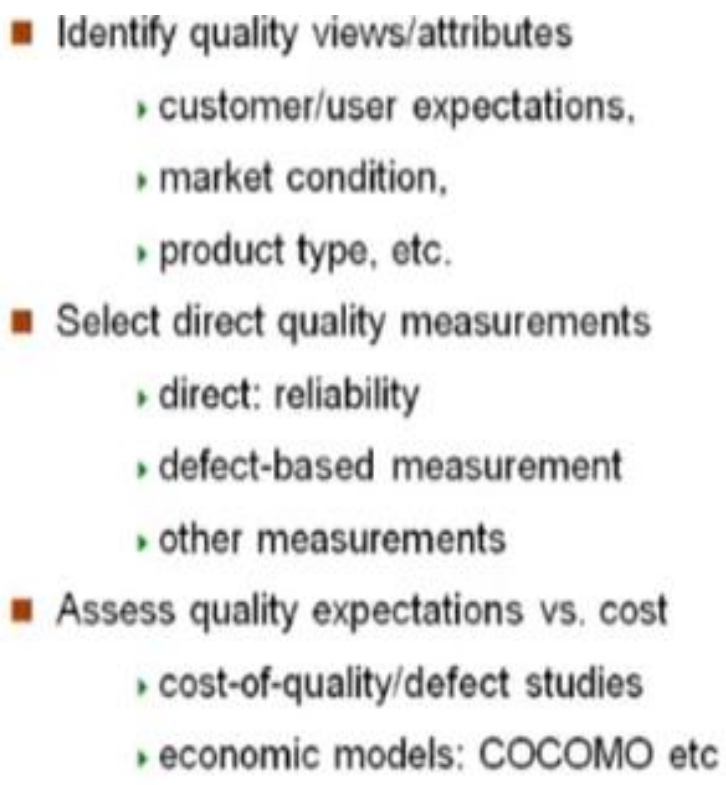
QUALITY ENGINEERING: ACTIVITIES AND PROCESS

- ***Pre-QA activities:***
  - **Quality planning**
    - These are the activities that should be carried out before carrying out the regular QA activities. There are two major types of pre-QA activities in quality planning, including:
      - Set specific quality goals.
      - Form an overall QA strategy, which includes two sub-activities:
        - Select appropriate QA activities to perform.
        - Choose appropriate quality measurements and models to provide feedback, quality assessment and improvement.

PRE-QA PLANNING



- Pre-QA planning:
  - Quality goal
  - Overall QA strategy:
  - QA activities to perform?
  - measurement/feedback planning
- Setting quality goal(s):
  - Identify quality views/attributes
  - Select direct quality measurements
  - Assess quality expectations vs. cost



- Identify quality views/attributes
  - customer/user expectations,
  - market condition,
  - product type, etc.
- Select direct quality measurements
  - direct: reliability
  - defect-based measurement
  - other measurements
- Assess quality expectations vs. cost
  - cost-of-quality/defect studies
  - economic models: COCOMO etc

QUALITY ENGINEERING: ACTIVITIES AND PROCESS

- ***In-QA activities***:
  - **Executing planned QA activities and handling discovered defects**
    - In addition to performing selected QA activities, an important part of this normal execution is to deal with the discovered problems.

QUALITY ENGINEERING: ACTIVITIES AND PROCESS

- ***Post-QA activities:***
  - **Quality measurement, assessment and improvement**
    - The primary purpose of these activities is to provide quality assessment and feedback so that various management decisions can be made and possible quality improvement initiatives can be carried out
- Short-term feedback
  - provides information for progress tracking, activity scheduling, and identification of areas that need special attentions
- Long-term feedback
  - Feedback to quality planning so that necessary adjustment can be made to quality goals and QA strategies
  - Feedback to the quality assessment and improvement activities

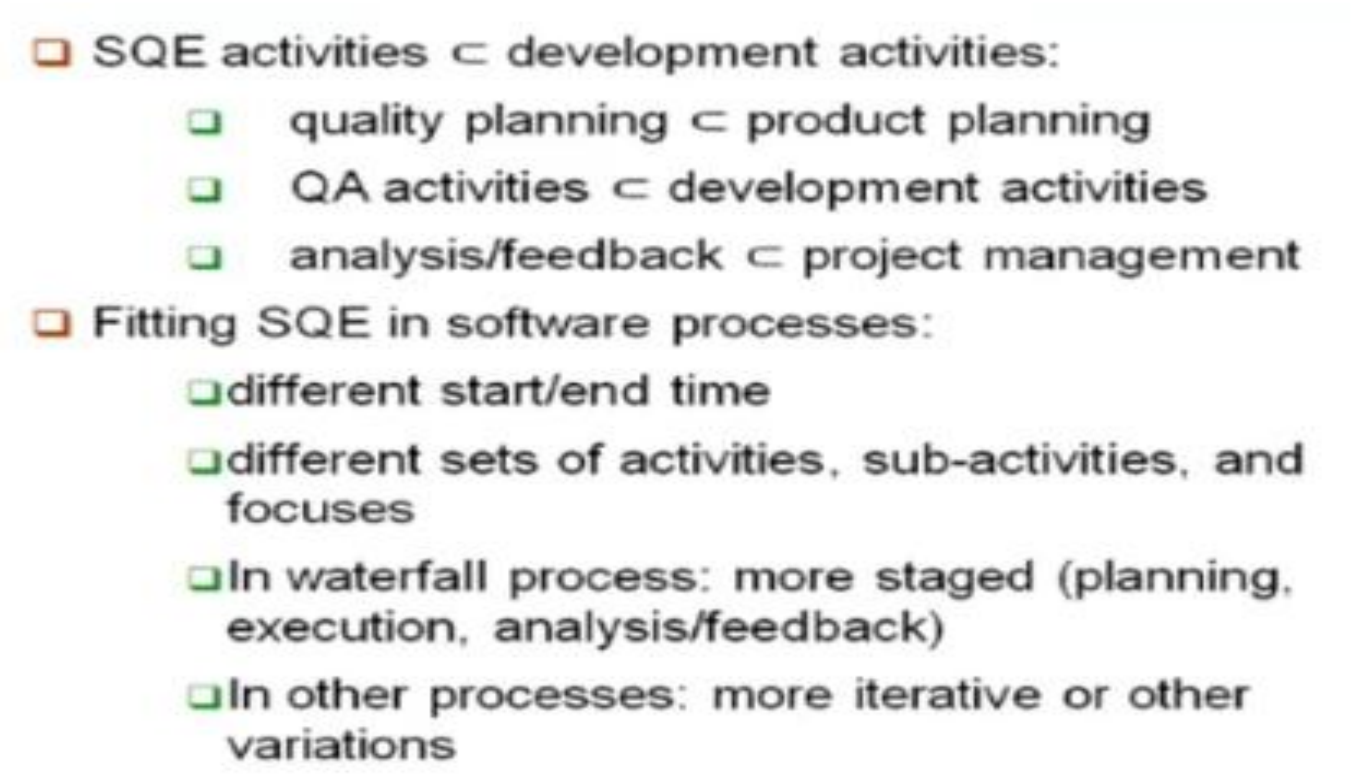
QUALITY ENGINEERING AND QUALITY IMPROVEMENT PARADIGM (QIP)

- The Quality Improvement Paradigm (QIP) suggests the following activities to achieve quality improvement
  - measurement, analysis, feedback, and organizational support
- QIP includes three interconnected steps: understanding, assessing, and packaging, which form a feedback and improvement loop, as briefly described below:
  - The first step is to understand the baseline so that improvement opportunities can be identified and clear measurable goals can be set. All future process changes are measured against this baseline.
  - The second step is to introduce process changes through experiments, pilot projects, assess their impact, and fine tune these process changes.
  - The last step is to package baseline data, experiment results, local experience, and updated process as the way to infuse the findings of the improvement program into the development organization.

QUALITY ENGINEERING IN SOFTWARE PROCESS

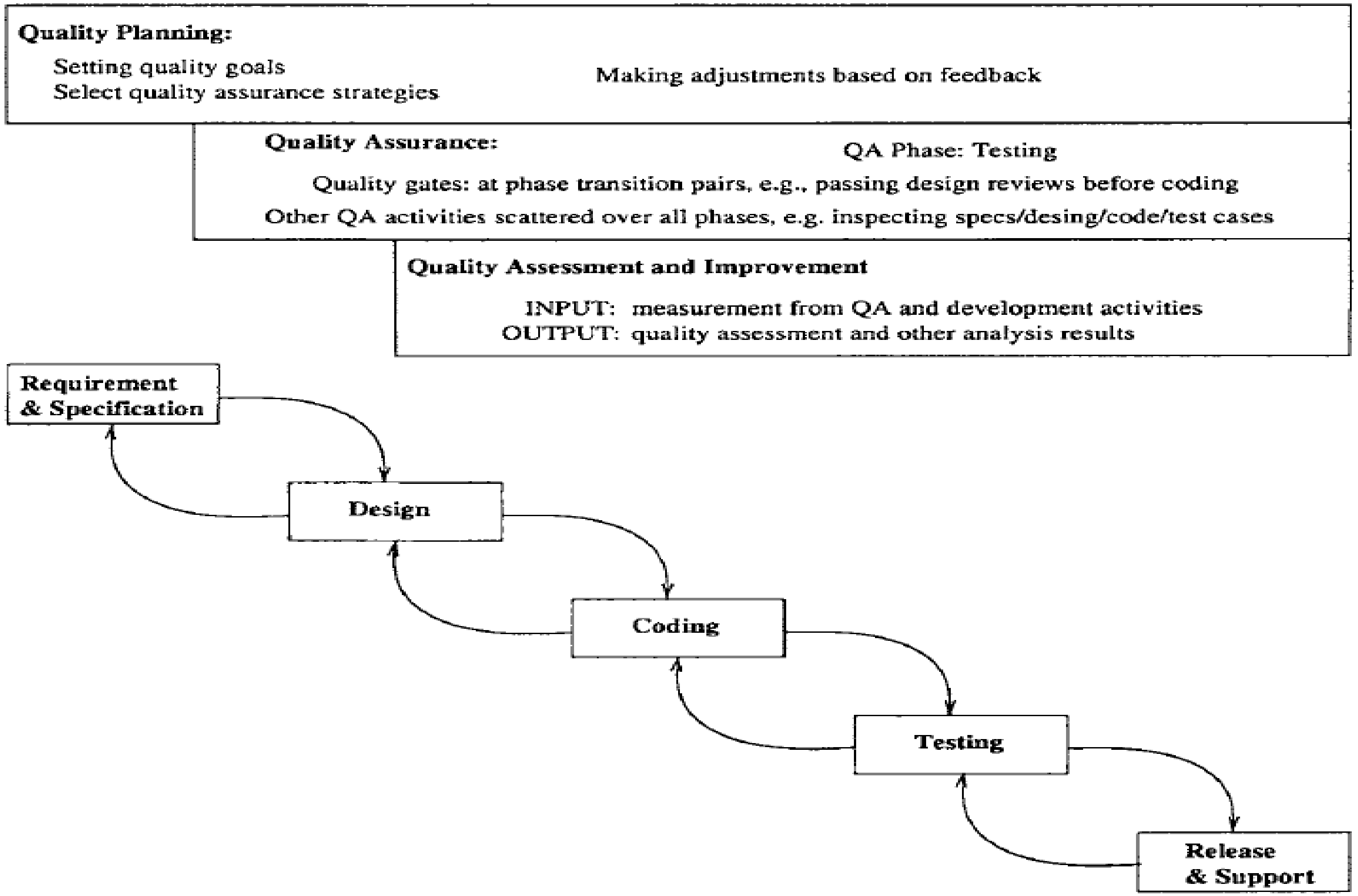
- The quality engineering process forms an integral part of the overall software engineering process, where other concerns, such as cost and schedule, are also considered and managed.
- Individual QA activities can be carried out and integrated into the software process. When we broaden our scope to quality engineering, it also covers pre-QA quality planning as well as the post-QA measurement and analysis activities carried out parallel to and after QA activities to provide feedback and other useful information.
- All these activities and the quality engineering process can be integrated into the overall software process.

QUALITY ENGINEERING IN SOFTWARE PROCESS



- SQE activities ⊂ development activities:
  - quality planning ⊂ product planning
  - QA activities ⊂ development activities
  - analysis/feedback ⊂ project management
- Fitting SQE in software processes:
  - different start/end time
  - different sets of activities, sub-activities, and focuses
  - In waterfall process: more staged (planning, execution, analysis/feedback)
  - In other processes: more iterative or other variations

QUALITY ENGINEERING IN THE WATERFALL PROCESS



Quality Planning:

Setting quality goals
Select quality assurance strategies

Making adjustments based on feedback

Quality Assurance:

QA Phase: Testing

Quality gates: at phase transition pairs, e.g., passing design reviews before coding

Other QA activities scattered over all phases, e.g. inspecting specs/desing/code/test cases

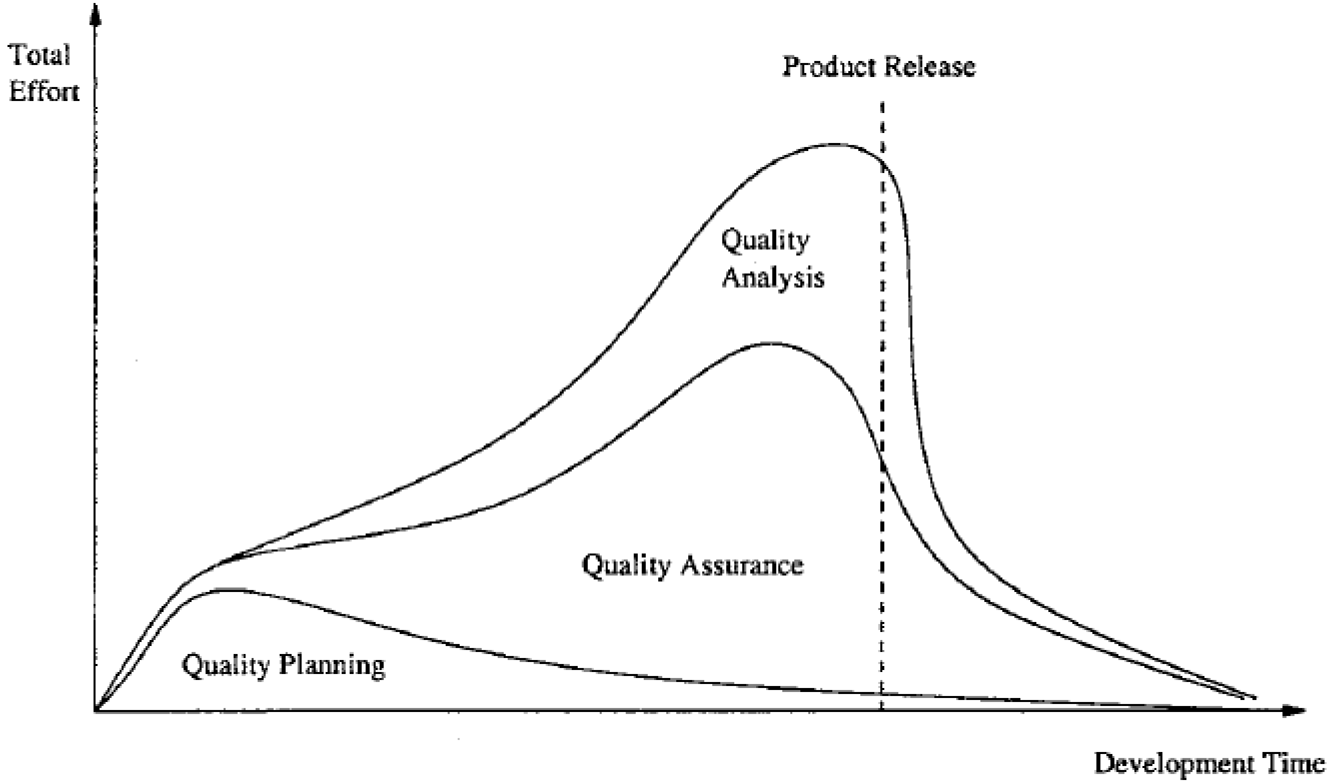
Quality Assessment and Improvement

INPUT: measurement from QA and development activities
OUTPUT: quality assessment and other analysis results

EFFORT PROFILE

- Among the three major types of activities in the quality engineering process, the execution of specific QA activities is central to dealing with defects and assuring quality for the software products. Therefore, they should and normally do consume the most resources in terms of human effort as well as utilization of computing and other related resources.
- However, the effort distribution among the three is not constant over time because of the process characteristics described above and the shifting focus over time.

EFFORT PROFILE



The exact profile based on real data would not be as smooth and would naturally show large amount of variability, with many small peaks and valleys. But the general shape and pattern should preserve.

EFFORT PROFILE

- Waterfall process would see more dominance of quality planning in the beginning, and dominance of testing near product release, and measurement and quality assessment activities peak right before product release.
- Other development processes, such as incremental, iterative, spiral, and extreme programming processes, would be associated with curves that vary less between the peaks and valleys. QA is spread out more evenly in these processes than in the waterfall process, although it is still expected to peak a little bit before product release.

CONCLUDING REMARKS

- To manage the quality assurance (QA) activities and to provide realistic opportunities of quantifiable quality improvement, we need to go beyond QA to perform the following:
  - Quality planning before specific QA activities are carried out, in the pre-QA activities in software quality engineering
  - Need to set the overall quality goal by managing customer's quality expectations under the project cost and budgetary constraints
- Quality quantification and improvement through measurement, analysis, feedback, and follow-up activities.
  - These activities need to be carried out after the start of specific QA activities, in the so-called post-QA activities in software quality engineering

That is all