

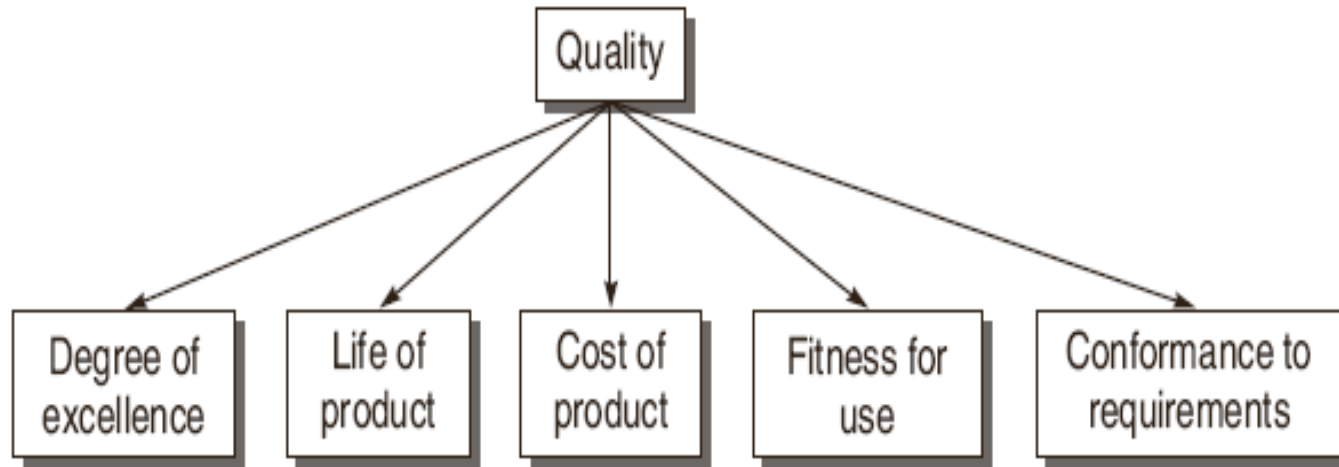


# **SOFTWARE QUALITY MANAGEMENT**

## **Module 5**

1

# Quality as Multidimensional Concept



The degree to which a product or service possesses a desired combination of attributes

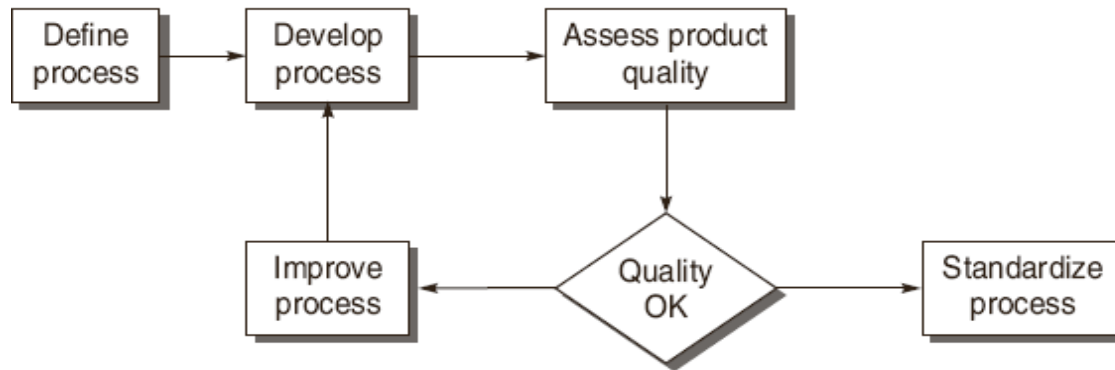
# WHAT IS QUALITY?

- “an inherent or distinguishing characteristic; a property; having a high degree of excellence”
- Features & functionality
  - “fitness for use”
  - “conformance to requirements”



# Broadening the Concept of Quality

- Product Quality
- Process Quality



**Figure 13.2** Process affects quality

# FIVE VIEWS OF SOFTWARE QUALITY

- Transcendental view
- User view
- Manufacturing view
- Product view
- Value-based view

# FIVE VIEWS OF SOFTWARE QUALITY

## ○ Transcendental view

- Quality is something that can be recognized through experience, but not defined in some tractable form.
- A good quality object stands out, and it is easily recognized.

## ○ User view

- Quality concerns the extent to which a product meets user needs and expectations.
- Is a product fit for use?
- This view is highly personalized.
  - A product is of good quality if it satisfies a large number of users.
  - It is useful to identify the product attributes which the users consider to be important.
- This view may encompass many subject elements, such as *usability*, *reliability*, and *efficiency*.

# FIVE VIEWS OF SOFTWARE QUALITY

## ○ Manufacturing view

- This view has its genesis in the manufacturing industry – auto and electronics.
- Key idea: Does a product satisfy the requirements?
  - Any deviation from the requirements is seen as reducing the quality of the product.
- The concept of process plays a key role.
- Products are manufactured “right the first time” so that the cost is reduced
  - Development cost
  - Maintenance cost
- Conformance to requirements leads to uniformity in products.
- Some argue that such uniformity does not guarantee quality.
- Product quality can be incrementally improved by improving the process.
  - The CMM and ISO 9001 models are based on the manufacturing view.

# FIVE VIEWS OF SOFTWARE QUALITY

## ○ Product view

- Hypothesis: If a product is manufactured with good internal properties, then it will have good external properties.
- One can explore the causal relationship between *internal properties* and *external qualities*.
- Example: *Modularity enables testability*.

## ○ Value-based view

- This represents the merger of two concepts: *excellence* and *worth*.
- Quality is a measure of excellence, and value is a measure of worth.
- Central idea
  - How much a customer is willing to pay for a certain level of quality.
  - Quality is meaningless if a product does not make economic sense.
  - The value-based view makes a trade-off between cost and quality.



# MCCALL'S QUALITY FACTORS AND CRITERIA

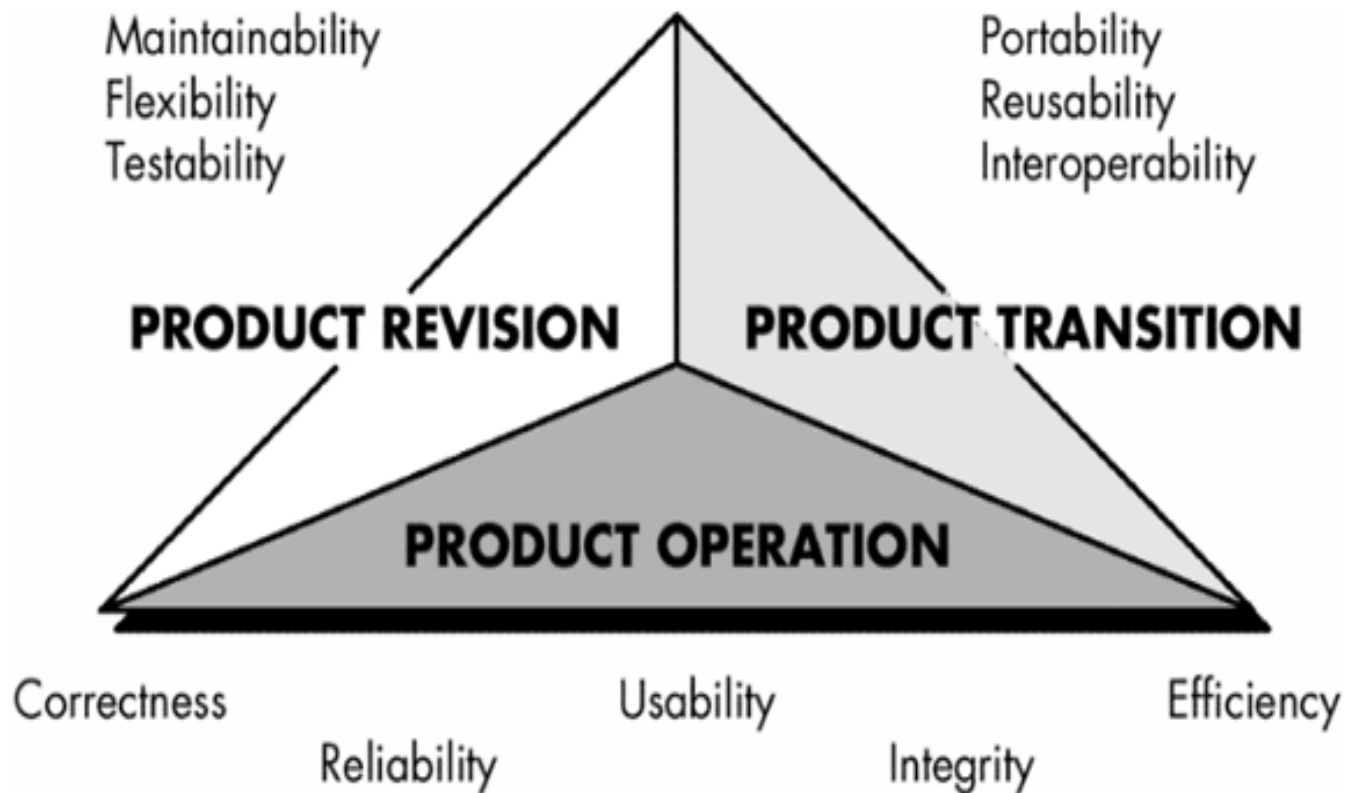
## ○ Quality Factors

- McCall, Richards, and Walters studied the concept of software quality in terms of two key concepts as follows:
  - *quality factors*, and
  - *quality criteria*.
- A quality factor represents the behavioral characteristic of a system.
  - Examples: correctness, reliability, efficiency, testability, portability, ...

Quality Factors	Definitions
<b>Correctness</b>	The extent to which a program satisfies its specifications and fulfills the user's mission objectives.
<b>Reliability</b>	The extent to which a program can be expected to perform its intended function with required precision.
<b>Efficiency</b>	The amount of computing resources and code required by a program to perform a function.
<b>Integrity</b>	The extent to which access to software or data by unauthorized persons can be controlled.
<b>Usability</b>	The effort required to learn, operate, prepare input, and interpret output of a program.
<b>Maintainability</b>	The effort required to locate and fix a defect in an operational program.
<b>Testability</b>	The effort required to test a program to ensure that it performs its intended functions.
<b>Flexibility</b>	The effort required to modify an operational program.
<b>Portability</b>	The effort required to transfer a program from one hardware and/ or software environment to another.
<b>Reusability</b>	The extent to which parts of a software system can be reused in other applications.
<b>Interoperability</b>	The effort required to couple one system with another.

Table : McCall's quality factors

# MCCALL'S QUALITY FACTORS



# MCCALL'S QUALITY FACTORS AND CRITERIA

Quality Categories	Quality Factors	Broad Objectives
<b>Product Operation</b>	Correctness Reliability Efficiency Integrity Usability	Does it do what the customer wants? Does it do it accurately all of the time? Does it quickly solve the intended problem? Is it secure? Can I run it?
<b>Product Revision</b>	Maintainability Testability Flexibility	Can it be fixed? Can it be tested? Can it be changed?
<b>Product Transition</b>	Portability Reusability Interoperability	Can it be used on another machine? Can parts of it be reused? Can it interface with another system?

# MCCALL'S QUALITY FACTORS AND CRITERIA

## ○ Quality Criteria

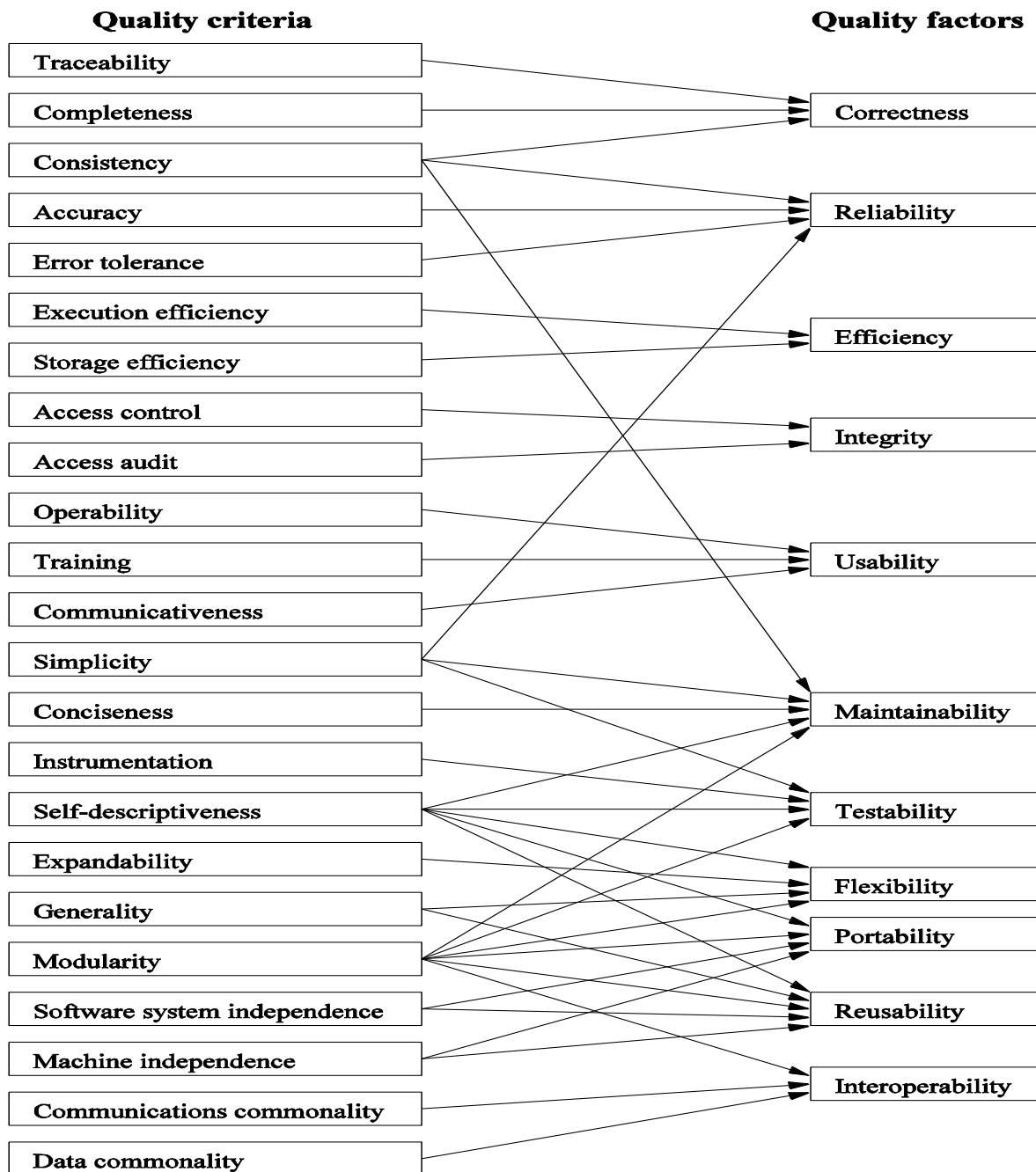
- A quality criterion is an attribute of a quality factor that is related to software development.
- Example:
  - Modularity is an attribute of the architecture of a software system.
  - A highly modular software allows designers to put cohesive components in one module, thereby increasing the maintainability of the system.
- In Table:- quality criteria have been listed.

<b>Quality Criteria</b>	<b>Definitions of Quality Criteria</b>
<b>Access audit</b>	The ease with which software and data can be checked for compliance with standards or other requirements.
<b>Access control</b>	The provisions for control and protection of the software and data.
<b>Accuracy</b>	The precisions of computations and output.
<b>Communication commonality</b>	The degree to which standard protocols and interfaces are used.
<b>Completeness</b>	The degree to which a full implementation of the required functionalities has been achieved.
<b>Communicativeness</b>	The ease with which inputs and outputs can be assimilated.
<b>Conciseness</b>	The compactness of the source code, in terms of lines of code.
<b>Consistency</b>	The use of uniform design and implementation techniques and notations throughout a project.
<b>Data commonality</b>	The use of standard data representations.
<b>Error tolerance</b>	The degree to which continuity of operation is ensured under adverse conditions.
<b>Execution efficiency</b>	The run-time efficiency of the software.
<b>Expandability</b>	The degree to which storage requirements or software functions can be expanded.
<b>Generality</b>	The breadth of the potential application of software components.
<b>Hardware independence</b>	The degree to which the software is dependent on the underlying hardware.
<b>Instrumentation</b>	The degree to which the software provides for measurements of its use or identification of errors.
<b>Modularity</b>	The provision of highly independent modules.
<b>Operability</b>	The ease of operation of the software.
<b>Self-documentation</b>	The provision of in-line documentation that explains the implementation of components.
<b>Simplicity</b>	The ease with which the software can be understood.
<b>Software system independence</b>	The degree to which the software is independent of its software environment – non-standard language constructs, operating system, libraries, database management system, etc.
<b>Software efficiency</b>	The run-time storage requirements of the software.
<b>Traceability</b>	The ability to link software components to requirements.
<b>Training</b>	The ease with which new users can use the system.

# MCCALL'S QUALITY FACTORS AND CRITERIA

## ○ Relationship Between Quality Factors and Quality Criteria

- Each quality factor is positively influenced by a set of quality criteria, and the same quality criterion impacts a number of quality factors.
  - Example: Simplicity impacts reliability, usability, and testability.
- If an effort is made to improve one quality factor, another quality factor may be degraded.
  - Portable code may be less efficient.
- Some quality factors positively impact others.
  - An effort to improve the correctness of a system will increase its reliability.



## McCALL'S QUALITY FACTORS AND CRITERIA



# QUALITY MANAGEMENT

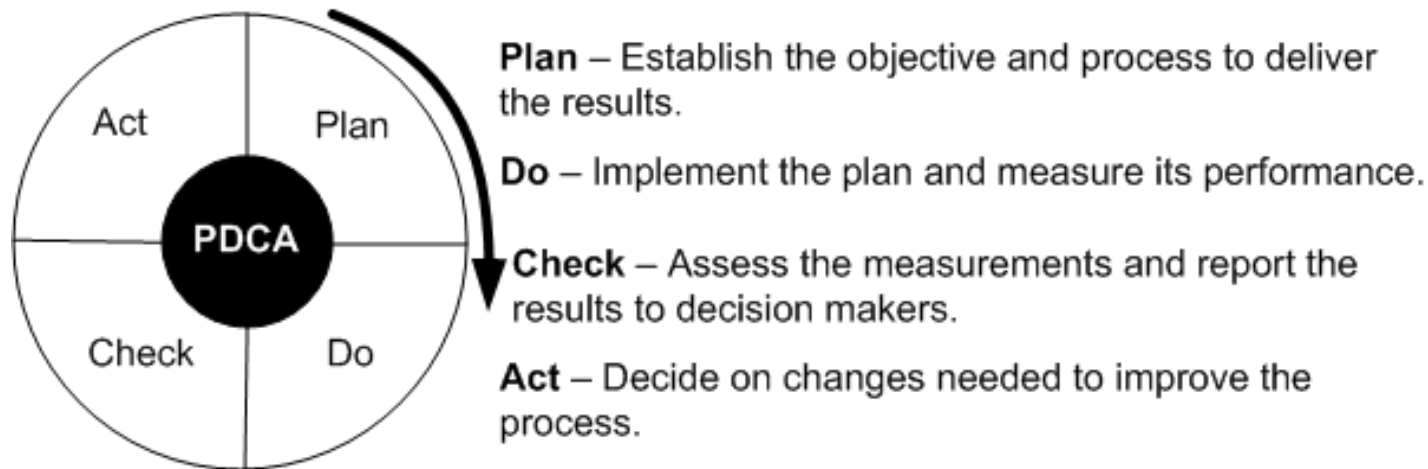
Quality management is to

- plan suitable Quality control and Quality assurance activities,
- define procedures and standards which should be used during software development and verifying that these are being followed by everyone
- Properly execute and control activities.
- Quality Management and Project Management



# THE QUALITY REVOLUTION - PDCA CYCLE

## The Shewhart cycle

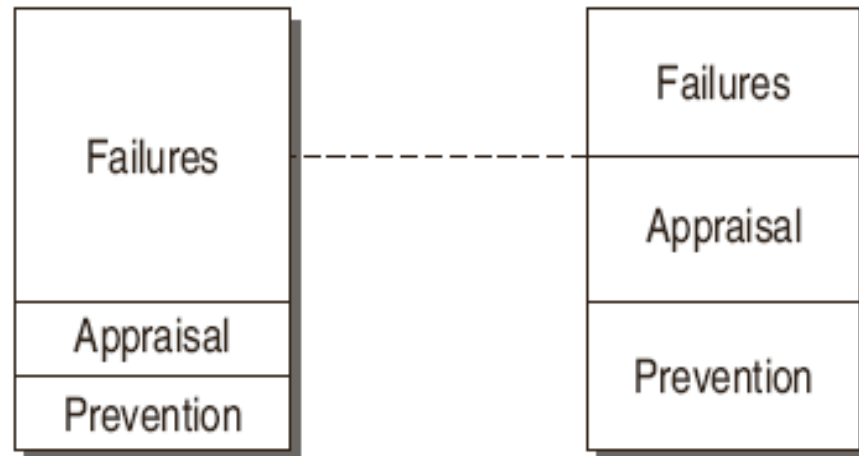


- Deming introduced Shewhart's PDCA cycle to Japanese researchers
- It illustrates the activity sequence:
  - Setting goals
  - Assigning them to measurable milestones
  - Assessing the progress against the milestones
  - Take action to improve the process in the next cycle



# QUALITY COST & BENEFITS OF INVESTMENT ON QUALITY

1. Prevention Costs
2. Appraisal Costs
3. Failure Costs
  - Internal Failure Costs
  - External Failure Costs



# QUALITY CONTROL AND QUALITY ASSURANCE

Quality Control is basically related to software product such that there is minimum variation, according to the desired specifications. This variation is checked at each step of development. Quality control may include the following activities: Reviews, Testing using manual techniques or with automated tools (V & V).

Quality Assurance is largely related to the process. In addition, quality assurance activities are in management zone. Therefore auditing and reporting of quality based on quantitative measurements are also performed.

# Methods of quality management



# PROCEDURAL APPROACH TO QM

- SQA activities
- SQA relationships with other assurance activities
- Configuration Management monitoring
- Verification and validation monitoring
- Formal test monitoring
- SQA during SDLC

# SOFTWARE QUALITY ASSURANCE BEST PRACTICE

- **Continuous improvement:** All the standard process in SQA must be improved **frequently** and made **official** so that the other can follow. This process should be **certified** by popular organization such as ISO, CMMI... etc.
- **Documentation:** All the QA policies and methods, which are defined by QA team, should be documented for training and reuse for future projects.
- **Experience:** Choosing the members who are seasoned SQA auditors is a good way to ensure the quality of management review
- **Tool Usage:** Utilizing tool such as the tracking tool, management tool for SQA process reduces SQA effort and project cost.
- **Metrics:** Developing and creating metrics to track the software quality in its current state, as well as to compare the improvement with previous versions, will help increase the value and maturity of the Testing process
- **Responsibility:** The SQA process is not the SQA member's task, but **everyone's** task. Everybody in the team is responsible for quality of product, not just the test lead or manager.

# QUANTITATIVE APPROACH TO QM

## Major Issues

- Setting Quality Goal

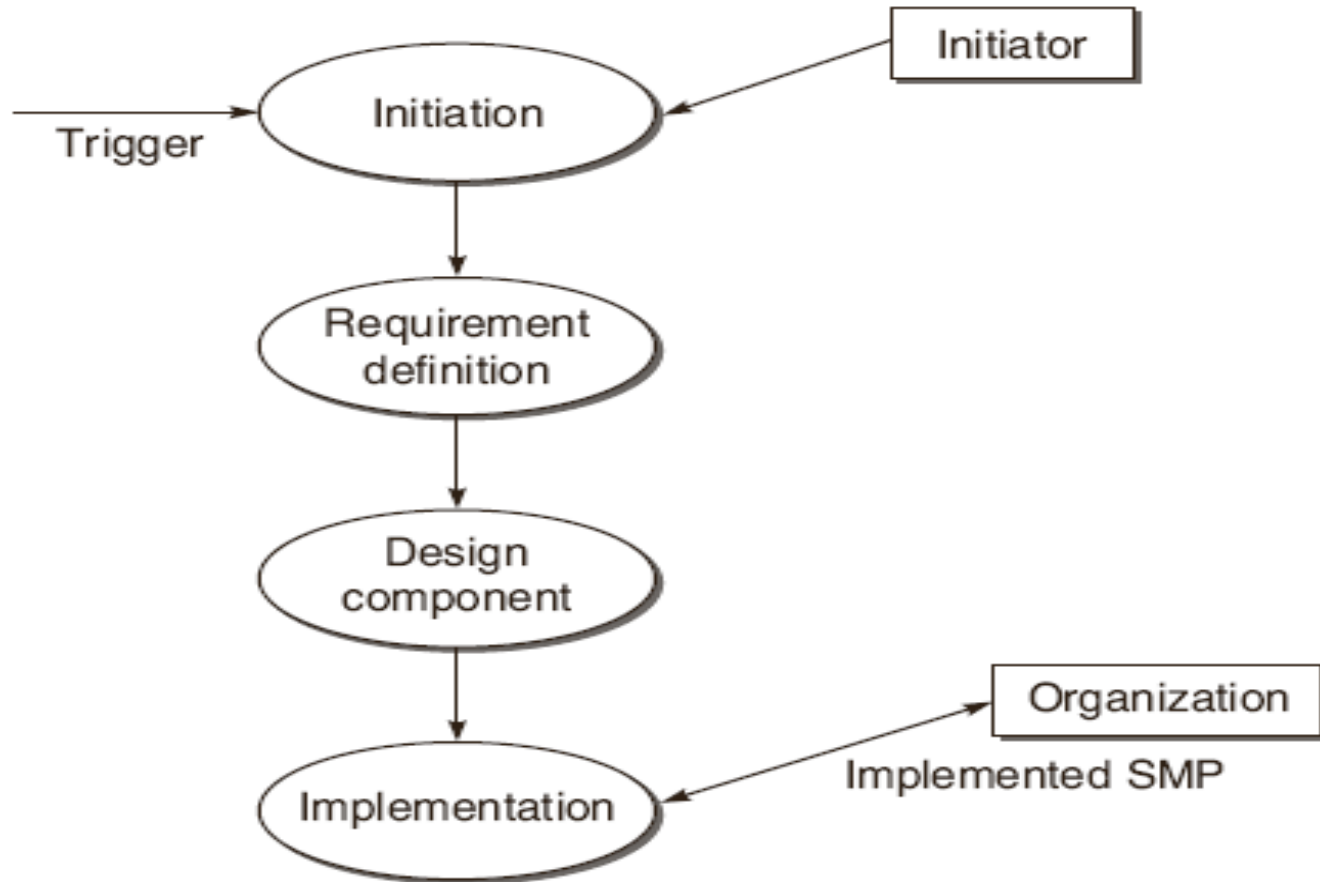
Estimate for defects(P)current Project=

Defects(SP)X effort estimate(P)/Actual effort(SP)

- Managing software development process quantitatively



# PAUL GOODMAN MODEL FOR SOFTWARE METRICS PROGRAM



# SOFTWARE QUALITY METRICS

## Product Quality Metrics

- **Mean time to failure (MTTF)**

MTTF metric is an estimate of the average or mean time until a product's first failure occurs.

## Defect Density Metrics

- **Defect Density = Number of Defects / Size of Product**

## Customer problem metrics

- problems per user month (PUM) = Total problems reported by the customer for a time period / Total number of licensed months of the software during the period.

- **Customer Satisfaction Metrics**



# SOFTWARE QUALITY METRICS

## In-Process Quality Metrics

- Defect Density during testing
- Defect Arrival pattern during Testing
- Defect Removal Efficiency

**DRE = (Defects removed during the month / No. of problems arrivals during the month) X 100**



# Software Quality Metrics

## Metrics for Software maintenance

### Fix backlog and backlog management index

**BMI = (Number of problems closed during the month /  
Number of problem arrivals during the month) x 100 %**

### Fix response time and fix responsiveness

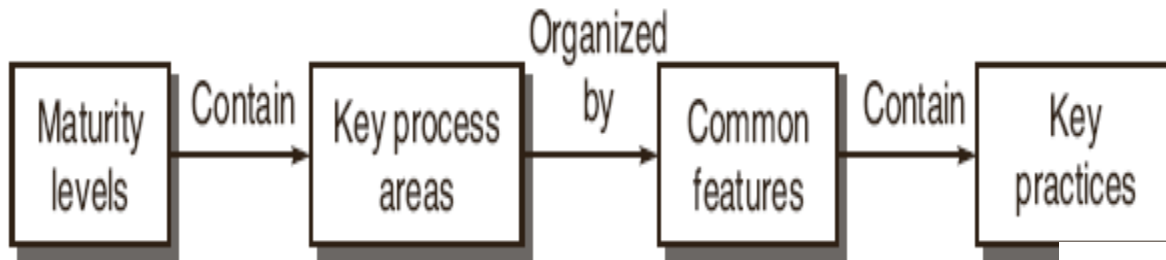
### Percent Delinquent fixes

**Percent Delinquent fixes = (Number of fixes that exceeded  
the response time criteria by severity level / Number of fixes  
delivered in a specified time) X 100%**

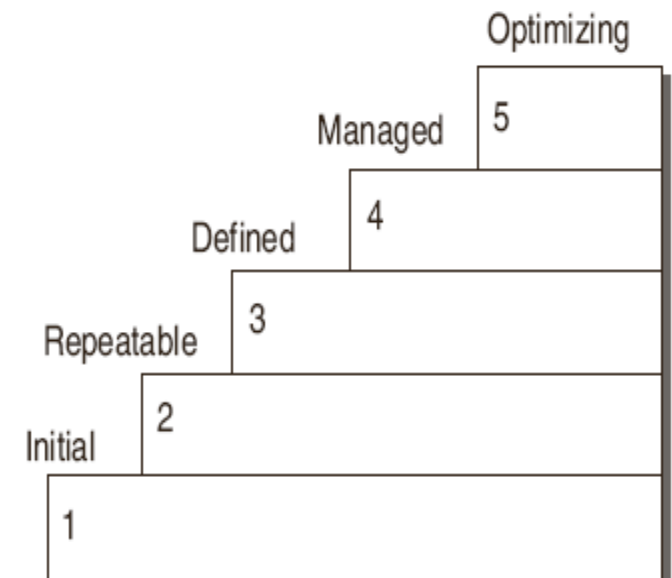
### Fix Quality: Defective fix percentage



# Capability Maturity Model (CMM)



Maturity levels	Indicate	Process capability
Key process areas	Achieve	Goals
Common features	Address	Implementation/Institutionalization
Key practices	Describe	Infrastructure/Activities



# CAPABILITY MATURITY MODEL (CMM)

**Table 13.2** KPAs for level 2

<i>KPAs, at this level, focus on the project's concerns related to establishing basic project management controls.</i>	
<b>KPA</b>	<b>Goals</b>
Requirement Management	Document the requirements properly. Manage the requirement changes properly.
Software Project Planning	Ensure proper project plan including estimation and listing of activities to be done.
Software Project Tracking and Oversight	Evaluate the actual performance of the project against the plans during project execution. Action, if there is deviation from plan.
Software Sub-contract Management	Selects qualified software sub-contractors. Maintain ongoing communications between prime contractor and the software sub-contractor. Track the software sub-contractor's actual results and performance against its commitments.
Software Quality Assurance	Plan the SQA activities. Ensure that there are proper processes by conducting review and audits. Take proper actions, if projects fail.
Software Configuration Management	Identify work products and documents to be controlled in the project. Control the changes in the software.

# CAPABILITY MATURITY MODEL (CMM)

**Table 13.3** KPAs for level 3

<i>KPAs, at this level, address both project and organizational issues, as the organization establishes an infrastructure that institutionalizes effective software engineering and management processes across all projects.</i>	
<b>KPA</b>	<b>Goals</b>
Organization Process Focus	Coordinate process development and improvement activities across the organization. Compare software processes with a process standard. Plan organization-level process development and improvement activities.
Organization Process Definition	Standard software processes are defined and documented.
Training Program	Identify training needs of various team members and implementation of training programs.
Integrated Software Management	Tailor the project from the standard defined process. Manage the project according to defined process.
Software Product Engineering	Define, integrate, and perform software engineering tasks. Keep the work products consistent especially if there are changes.
Inter-group Coordination	Ensure coordination between different groups.
Peer Reviews	Plan peer review activities. Identify and remove defects in the software work products.

# CAPABILITY MATURITY MODEL (CMM)

Table 13.4 KPAs for level 4

<i>KPAs, at this level, focus on establishing a quantitative understanding of both the software process and the software work products being built.</i>	
<i>KPA</i>	<i>Goals</i>
Quantitative Process Management	Plan the quantitative process management activities. Control the performance of the project's defined software process quantitatively. The process capability of the organization's standard software process is known in quantitative terms.
Software Quality Management	Set and plan quantitative quality goals for the project. Measure the actual performance of the project with quality goals and compare them with plans.



# CAPABILITY MATURITY MODEL (CMM)

Table 13.5 KPAs for level 5

<i>KPAs, at this level, cover the issues that both the organization and the projects must address to implement continuous and measurable software process improvement.</i>	
<i>KPA</i>	<i>Goals</i>
Defect Prevention	Plan the defect prevention activities. Identify, prioritize, and eliminate the common causes of bugs.
Technology Change Management	Plan the technology changes to be incorporated in the project, if any. Evaluate the effect of new technologies on quality and productivity.
Process Change Management	Plan the process improvement activities such that an organization-wide participation is there. Measure the change of improvement in the process.



## 6-SIGMA

- Originated by Motorola in Schaumburg, IL
- Based on competitive pressures in 1980s – “Our quality stinks”

<b>Sigma</b>	<b>Defects Per Million</b>
1 $\delta$	690,000
2 $\delta$	308,537
3 $\delta$	66,807
4 $\delta$	6,210
5 $\delta$	233
6 $\delta$	3.4



# 6-SIGMA

3 $\delta$	6 $\delta$
Five short or long landings at any major airport	One short or long landing in 10 years at all airports in the US
Approximately 1,350 poorly performed surgical operations in one week	One incorrect surgical operation in 20 years
Over 40,500 newborn babies dropped by doctors or nurses each year	Three newborn babies dropped by doctors or nurses in 100 years
Drinking water unsafe to drink for about 2 hours each month	Water unsafe to drink for one second every six years



# 6-SIGMA D-M-A-I-C CYCLE

## ○ Define

- The first step is to **define customer satisfaction goals and subgoals; for example, reduce cycle time, costs, or defects**. These goals then provide a baseline or benchmark for the process improvement.

## ○ Measure

- The Six Sigma team is responsible for **identifying a set of *relevant* metrics**.

## ○ Analyze

- With data in hand, the team can **analyze the data for trends, patterns, or relationships**. Statistical analysis allows for testing hypotheses, modeling, or conducting experiments.

## ○ Improve

- Based on **solid evidence**, improvements can be proposed and implemented. The Measure-Analyze-Improve steps are generally **iterative** to achieve target levels of performance.

## ○ Control

- Once target levels of performance are achieved, control methods and tools are put into place in order to **maintain performance**.



# 6-SIGMA ROLES & RESPONSIBILITIES

- ◉ Master black belts
  - People within the organization who have the highest level of technical and organizational experience and expertise. Master black belts train black.
- ◉ Black belts
  - Should be technically competent and held in high esteem by their peers. They are actively involved in the Six Sigma change process.
- ◉ Green belts
  - Are Six Sigma team leaders or project managers. Black belts generally help green belts choose their projects, attend training with them, and then assist them with their projects once the project begins.
- ◉ Champions
  - Leaders who are committed to the success of the Six Sigma project and can ensure that barriers to the Six Sigma project are removed. Usually a high-level manager who can remove obstacles that may involve funding, support, bureaucracy, or other issues that black belts are unable to solve on their own.



# SOFTWARE TOTAL QUALITY MANAGEMENT (STQM)

TQM is defined as a quality-centered, customer-focused, fact-based, team-driven, senior-management-led process to achieve an organization's strategic imperative through continuous process improvement.

T = Total = everyone in the organization

Q = Quality = customer satisfaction

M = Management = people and processes

# SOFTWARE TOTAL QUALITY MANAGEMENT (STQM)

## The elements of TQM / STQM

- Customer-focus/Customer-focus in software development
- Process / Process, technology, and development quality
- Human-side of quality/Human-side of software quality
- Measurement and analysis

# ISO 9000 STANDARD

The ISO 9000 family of quality management systems standards is designed to help organizations ensure that they meet the needs of customers and other stakeholders while meeting statutory and regulatory requirements related to a product or program. ISO 9000 deals with the fundamentals of quality management systems, including the seven quality management principles upon which the family of standards is based. ISO 9001 deals with the requirements that organizations wishing to meet the standard must fulfill.



# ISO 9000:2015 SOFTWARE QUALITY STANDARD AND FUNDAMENTALS

This International Standard provides the fundamental concepts, principles and vocabulary for quality management systems (QMS) and provides the foundation for other QMS standards. This International Standard is intended to help the user to understand the fundamental concepts, principles and vocabulary of quality management, in order to be able to effectively and efficiently implement a QMS and realize value from other QMS standards.

This International Standard proposes a well-defined QMS, based on a framework that integrates established fundamental concepts, principles, processes and resources related to quality, in order to help organizations realize their objectives. It is applicable to all organizations, regardless of size, complexity or business model. Its aim is to increase an organization's awareness of its duties and commitment in fulfilling the needs and expectations of its customers and interested parties, and in achieving satisfaction with its products and services.

# ISO 9000:2015 SOFTWARE QUALITY STANDARD AND FUNDAMENTALS

This International Standard describes the fundamental concepts and principles of quality management which are universally applicable to the following:

- organizations seeking sustained success through the implementation of a quality management system;
- customers seeking confidence in an organization's ability to consistently provide products and services conforming to their requirements;
- organizations seeking confidence in their supply chain that product and service requirements will be met;
- organizations and interested parties seeking to improve communication through a common understanding of the vocabulary used in quality management;
- organizations performing conformity assessments against the requirements of ISO 9001;
- providers of training, assessment or advice in quality management;
- developers of related standards.

# ISO 9000:2015 SOFTWARE QUALITY STANDARD AND FUNDAMENTALS

Essentially the layout of the standard is similar to the previous ISO standard in that it follows the Plan, Do, Check, Act cycle in a process based approach, but is now further encouraging this to have risk based thinking. The purpose of the quality objectives is to determine the conformity of the requirements (customers and organizations), facilitate effective deployment and improve the quality management system

# ISO 9000:2015 SOFTWARE QUALITY STANDARD AND FUNDAMENTALS

- Risk-based thinking is essential for achieving an effective quality management system. The concept of risk-based thinking has been implicit in previous editions of this International Standard including, for example, carrying out preventive action to eliminate potential nonconformities, analysing any nonconformities that do occur, and taking action to prevent recurrence that is appropriate for the effects of the nonconformity.
- To conform to the requirements of this International Standard, an organization needs to plan and implement actions to address risks and opportunities. Addressing both risks and opportunities establishes a basis for increasing the effectiveness of the quality management system, achieving improved results and preventing negative effects.
- Opportunities can arise as a result of a situation favourable to achieving an intended result, for example, a set of circumstances that allow the organization to attract customers, develop new products and services, reduce waste or improve productivity. Actions to address opportunities can also include consideration of associated risks. Risk is the effect of uncertainty and any such uncertainty can have positive or negative effects. A positive deviation arising from a risk can provide an opportunity, but not all positive effects of risk result in opportunities.

# ISO 9000:2015 SOFTWARE QUALITY STANDARD AND FUNDAMENTALS

The ISO 9000 series are based on seven quality management principles (QMP)

QMP 1 – Customer focus

QMP 2 – Leadership

QMP 3 – Engagement of people

QMP 4 – Process approach

QMP 5 – Improvement

QMP 6 – Evidence-based decision making

QMP 7 – Relationship management

# ISO 9001:2015 REQUIREMENTS

ISO 9001:2015 Quality management systems — Requirements is a document of approximately 30 pages which is available from the national standards organization in each country.

ISO 9001:2015 is the latest revision of the ISO 9001 standard. In it, there are 10 sections (clauses) with supporting subsections (sub clauses). The requirements to be applied to your quality management system (QMS) are covered in sections 4-10. To successfully implement ISO 9001:2015 within your organization, you must satisfy the requirements within clauses 4-10.

# ISO 9001:2015 REQUIREMENTS

Some of the key changes include:

High Level Structure of 10 clauses is implemented. Now all new standard released by ISO will have this High level structure.

- Greater emphasis on building a management system suited to each organization's particular needs
- A requirement that those at the top of an organization be involved and accountable, aligning quality with wider business strategy
- Risk-based thinking throughout the standard makes the whole management system a preventive tool and encourages continuous improvement
- Less prescriptive requirements for documentation: the organization can now decide what documented information it needs and what format it should be in
- Alignment with other key management system standards through the use of a common structure and core text
- Inclusion of Knowledge Management principles
- Quality Manual & Management representative is now not mandatory requirements.

# ISO 9001:2015 REQUIREMENTS

Contents of ISO 9001:2015 are as follows:

Section 1: Scope

Section 2: Normative references

Section 3: Terms and definitions

Section 4: Context of the organization

Section 5: Leadership

Section 6: Planning

Section 7: Support

Section 8: Operation

Section 9: Performance evaluation

Section 10: Improvement



# SOFTWARE QUALITY TOOLS

- Ishikawa Diagram
- Check List
- Control Chart
- Flow Chart
- Pareto Chart
- Histogram

# CAUSE & EFFECT DIAGRAMS

Kaoru Ishikawa (1915 - 1989)

- Studied under Deming
- Believed quality is a continuous process that relies on all levels of the organization

Cause and effect diagrams (Ishikawa Diagram) are used for understanding organizational or business problem causes.

Organizations face problems everyday and it is required to understand the causes of these problems in order to solve them effectively. Cause and effect diagrams exercise is usually a teamwork.

A brainstorming session is required in order to come up with an effective cause and effect diagram.

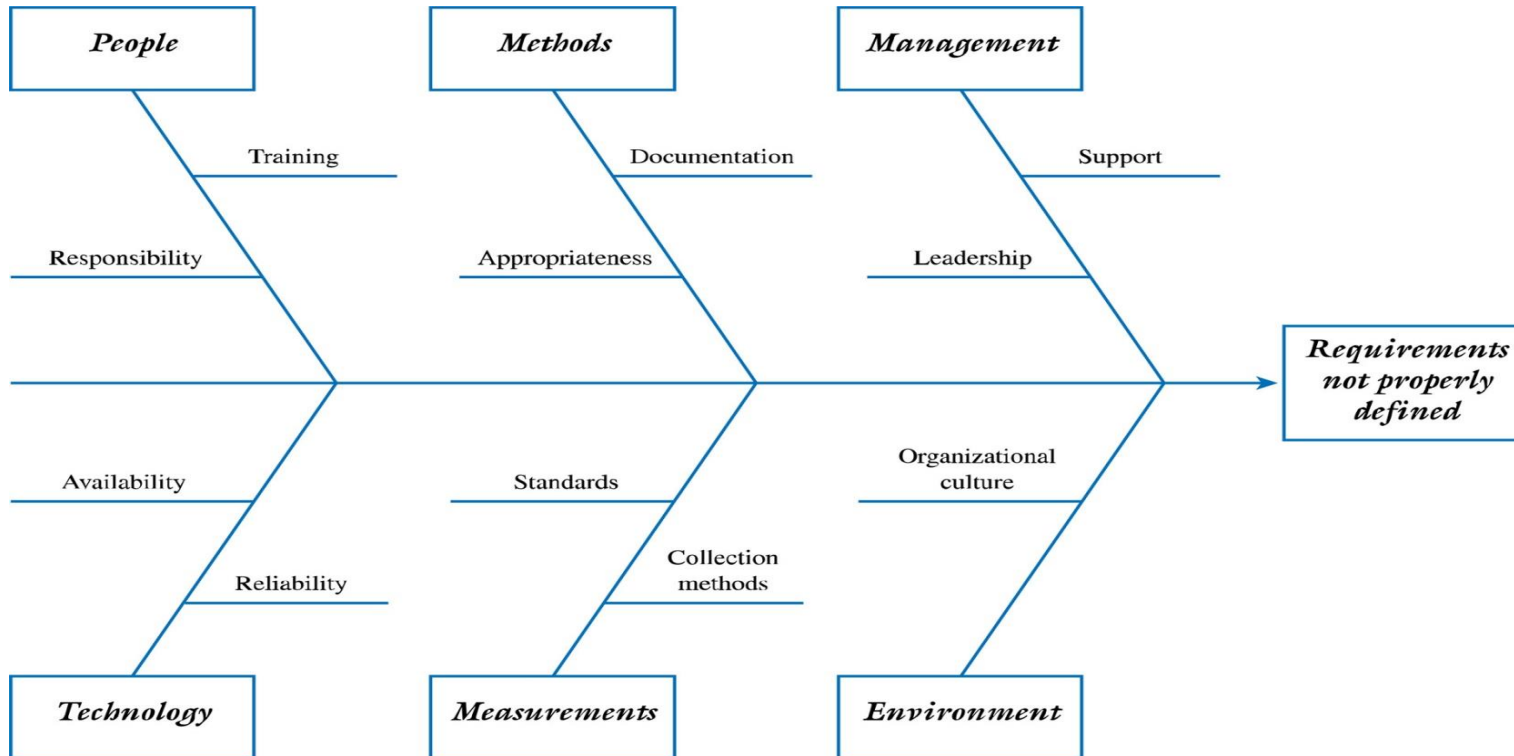
All the main components of a problem area are listed and possible causes from each area is listed.

Then, most likely causes of the problems are identified to carry out further analysis.



# ISHIKAWA, OR FISHBONE DIAGRAM

## BEST DEVELOPED BY BRAINSTORMING OR BY USING A LEARNING CYCLE APPROACH



# CONTROL CHARTS

- Walter A. Shewhart (1891 – 1967)
  - Worked for Western Electric Company (Bell Telephones)
  - Introduced the concept of the control chart as a tool better to understand variation and to allow management to shift its focus away from inspection and more towards the prevention of problems and the improvement of processes.



# CONTROL CHARTS

Common and special causes are the two distinct origins of variation in a process, as defined in the statistical thinking and methods of Walter A. Shewhart and W. Edwards Deming.

Briefly, "common causes", also called Natural patterns, are the usual, historical, quantifiable variation in a system, while "special causes" are unusual, not previously observed, non-quantifiable variation.



# CONTROL CHARTS

## Common causes

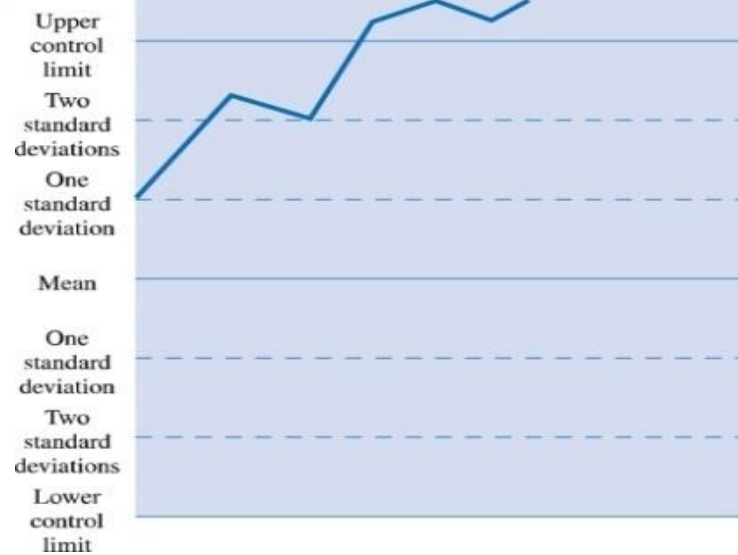
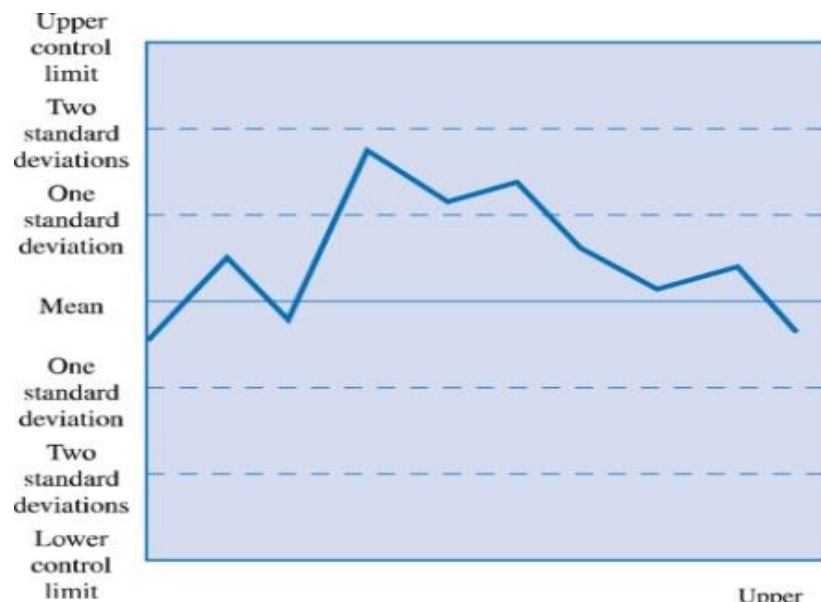
- Inappropriate procedures
- Poor design
- Poor maintenance of machines
- Measurement error
- Quality control error

## Special causes

- Faulty controllers
- Machine malfunction
- Computer crash
- Poor batch of raw material



# CONTROL CHARTS



# CONTROL CHARTS

## Purpose:

The primary purpose of a control chart is to predict expected product outcome.

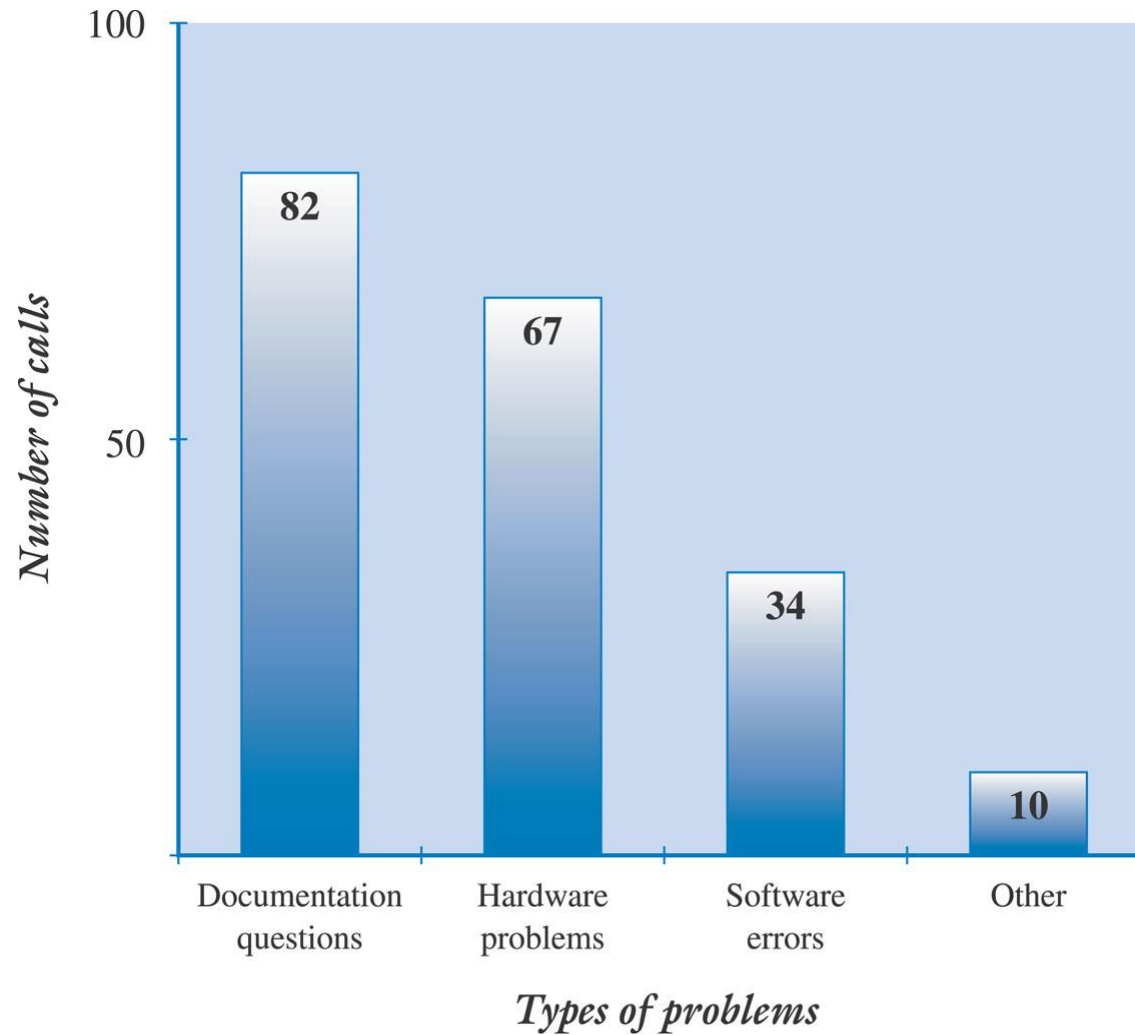
## Benefits:

- Predict process out of control and out of specification limits
- Distinguish between specific, identifiable causes of variation
- Can be used for statistical process control





# PARETO CHART



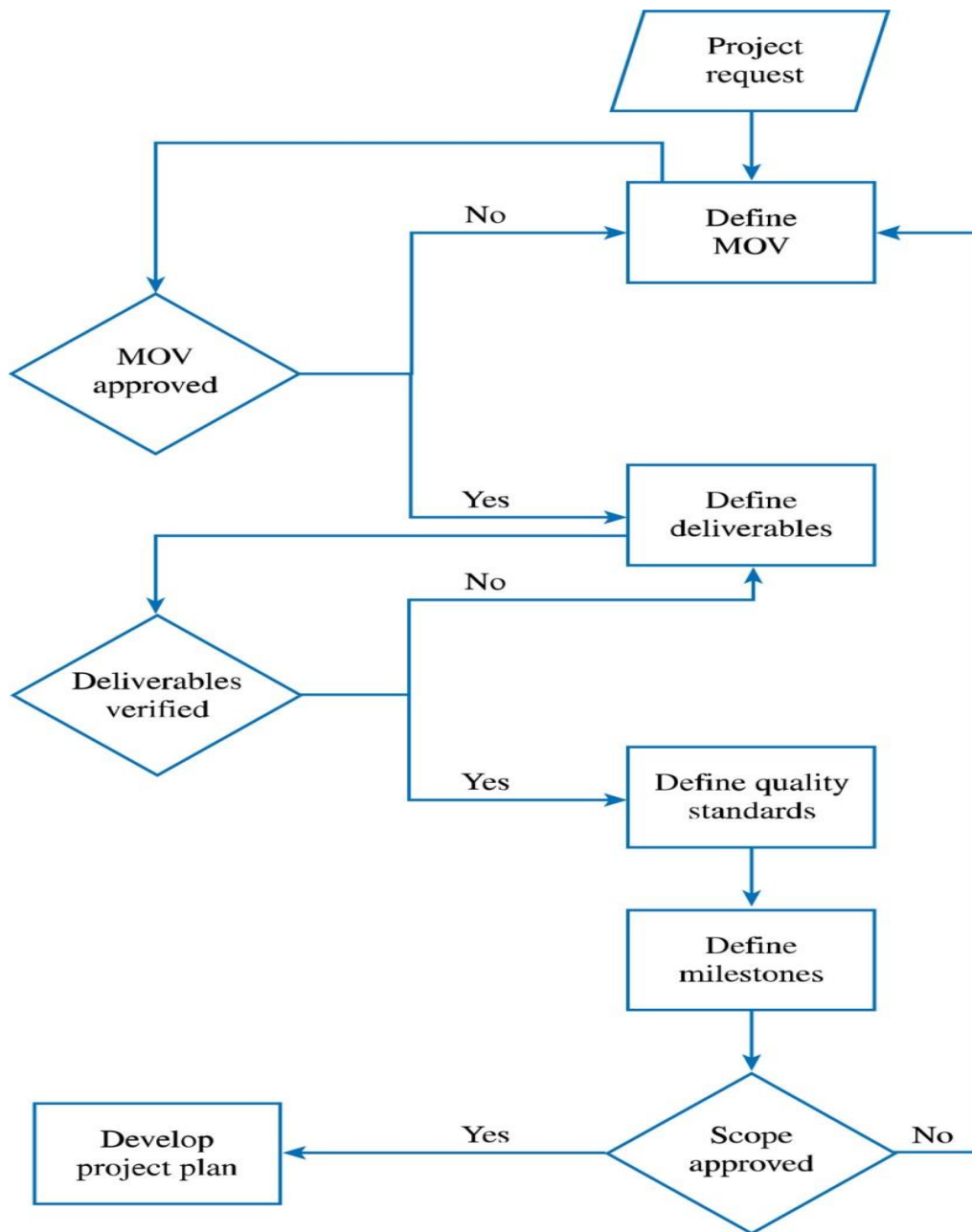
# PARETO CHART

Pareto charts are used for identifying a set of priorities. You can chart any number of issues/variables related to a specific concern and record the number of occurrences.

This way you can figure out the parameters that have the highest impact on the specific concern.

This helps you to work on the propriety issues in order to get the condition under control.






## FLOW CHART FOR PROJECT SCOPE VERIFICATION



# CHECK LIST

 <b>Check Sheet</b>								
Project Name:								
Project Manager:						Date:		
Defect/Issue	Sun	Mon	Tues	Wed	Thurs	Fri	Sat	Total
								0
								0
								0
								0
								0
								0
								0
								0
<b>Total</b>	0	0	0	0	0	0	0	0

# CHECK LIST/SHEET

A check sheet can be introduced as the most basic tool for quality. A check sheet is basically used for gathering and organizing data.

When this is done with the help of software packages such as Microsoft Excel, you can derive further analysis graphs and automate through macros available.

Therefore, it is always a good idea to use a software check sheet for information gathering and organizing needs.

One can always use a paper-based check sheet when the information gathered is only used for backup or storing purposes other than further processing.

# HISTOGRAM

Histogram is used for illustrating the frequency and the extent in the context of two variables.

Histogram is a chart with columns. This represents the distribution by mean. If the histogram is normal, the graph takes the shape of a bell curve.

If it is not normal, it may take different shapes based on the condition of the distribution. Histogram can be used to measure something against another thing.

# HISTOGRAM

