

Software Testing and Quality Assurance

Unit 6: Software Quality Assurance



Unit-6 Topics

❑ Software Quality Assurance

- Quality factors and criteria
- Relationship between quality factors and quality criteria
- Quality Metrics
- Quality Characteristics
- Software Quality Standard



What is Quality....?

The degree to which a system, component, or process **meets specified requirements**

(IEEE Standard Glossary)

Quality is conformance to specifications

(British Defense Industries Quality Assurance Panel)

The degree to which a system, component, or process **meets customer or user needs or expectations.**

(IEEE Standard Glossary)

Quality is conformance to requirements

(Philip Crosby)



What is Quality....?

Quality is synonymous with customer needs and expectations
(R J Mortiboys)

The totality of features and characteristics of a product or a service that bear on its ability to satisfy stated or implied needs

ISO 8402 definition of QUALITY

Quality is meeting the (stated) requirements of the customer- now and in the future
(Mike Robinson)

1. Functionality
2. Reliability
3. Usability
4. Efficiency
5. Maintainability
6. Portability

ISO 9216 Model: Major Quality characteristics



Problems with Software Quality

- Software specifications are usually *incomplete* and often *inconsistent*
- Some **GAPs** between:
 - customer quality requirements (efficiency, reliability etc.,
 - developer quality requirements (maintainability, reusability, etc.)
- Some quality requirements are *hard to specify* in an unambiguous way
 - directly measurable qualities (e.g., errors/KLOC),
 - indirectly measurable qualities (e.g., usability).

Quality management is not just about reducing defects!



Five views of software quality

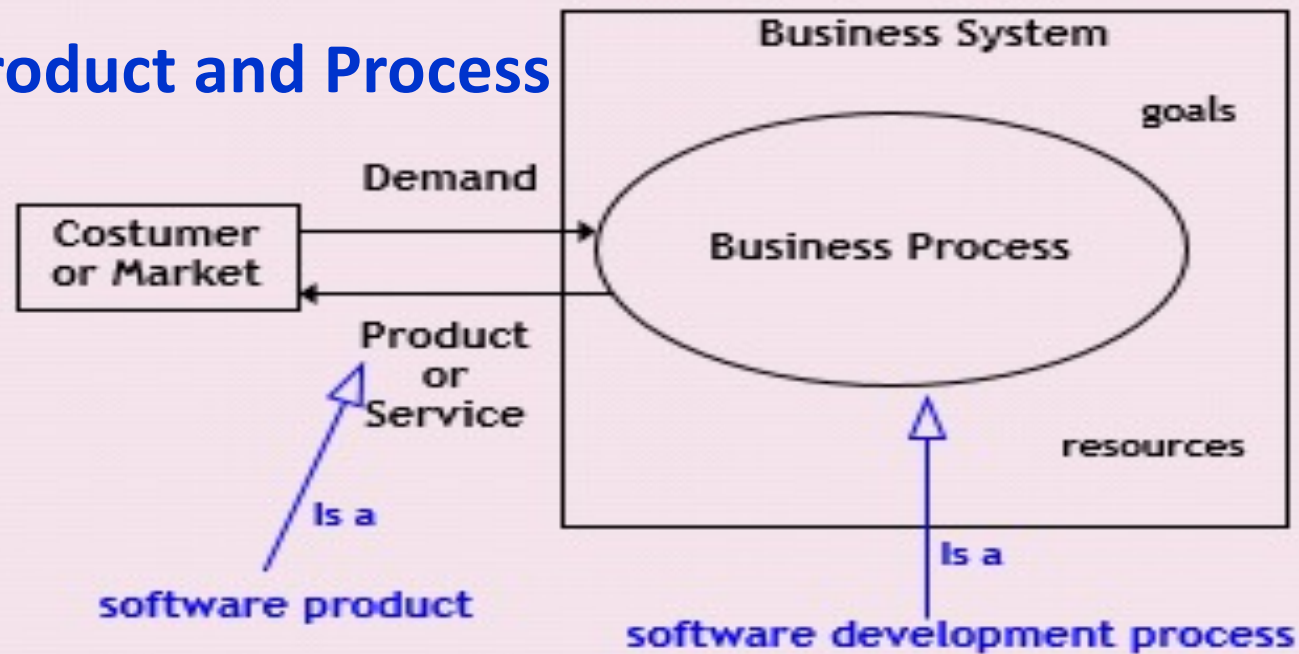
1. **Transcendental view:** Quality is something that can be recognized **through experience** but is not defined in some tractable form. **Quality is viewed to be something ideal.** In this case, no effort is made to express it using concrete measures. It is easily recognized.
2. **User view:** In this view, the extent to which a **product meets user needs and expectations.** The user is concerned with whether or not the product is fit for use. The product should meet user needs and expectations.
3. **Manufacturing view:** In this view, quality is seen as conforming to requirements. Any deviation from the stated requirements is seen **as reducing the quality of the product.** The concept of **process plays** a key role in the manufacturing view. Products are to be manufactured “right the first time” so that development cost and maintenance costs are reduced.
4. **Product view:** if a product is manufactured with good **internal characteristics** then it will have **good external quality.** The product view is attractive because it gives rise to an opportunity to explore causal relationships between *internal properties* and *external qualities* of a product.
5. **Value-Based view:** The value-based view represents a merger of two independent concepts: *excellence* and *worth*. Quality is a measure of excellence, and value is a measure of worth. The central view in this case is **how much a customer is willing to pay for a certain level of quality.**



Quality Management

Managing the quality of the software **process** and **products**

Product and Process



Process and product quality

- The quality of a developed product **is influenced** by the quality of the production process
- Particularly important in software development as some product quality attributes are hard to assess
- However, there is a very complex and poorly understood relationship between software processes and product quality



What is a software product?

Software product = computer programs (sources and executables) + associated documentation

Software products may be

1. **Custom** - developed for a **particular customer**, according to its specifications
2. **Generic** (“package”) - developed for a **general market**, to be sold to a range of different customers

Types of software products

Business support software

Includes software engineering tools in the software engineering business

Personal productivity software

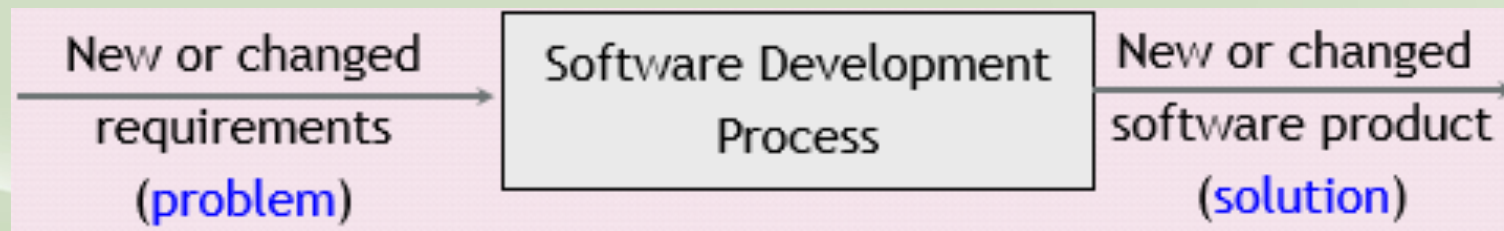
spreadsheets, word processing tools, ...

Embedded software



What is a software development process...?

- a set of activities whose **goal is** the **development or evolution of a software product**
 - To be followed/instantiated in individual software development projects
- It's the **main business process in a software development business**
- **Generic activities in all software processes are:**
 - **Specification** - what the system should do and its development constraints
 - **Development** - production of the software system
 - **Validation** - checking that the software is what the customer wants
 - **Evolution** - changing the software in response to changing demands



What is product quality?

The software product should deliver the **required functionality** (*functional requirements*) with the **required quality attributes** (*non-functional requirements*)

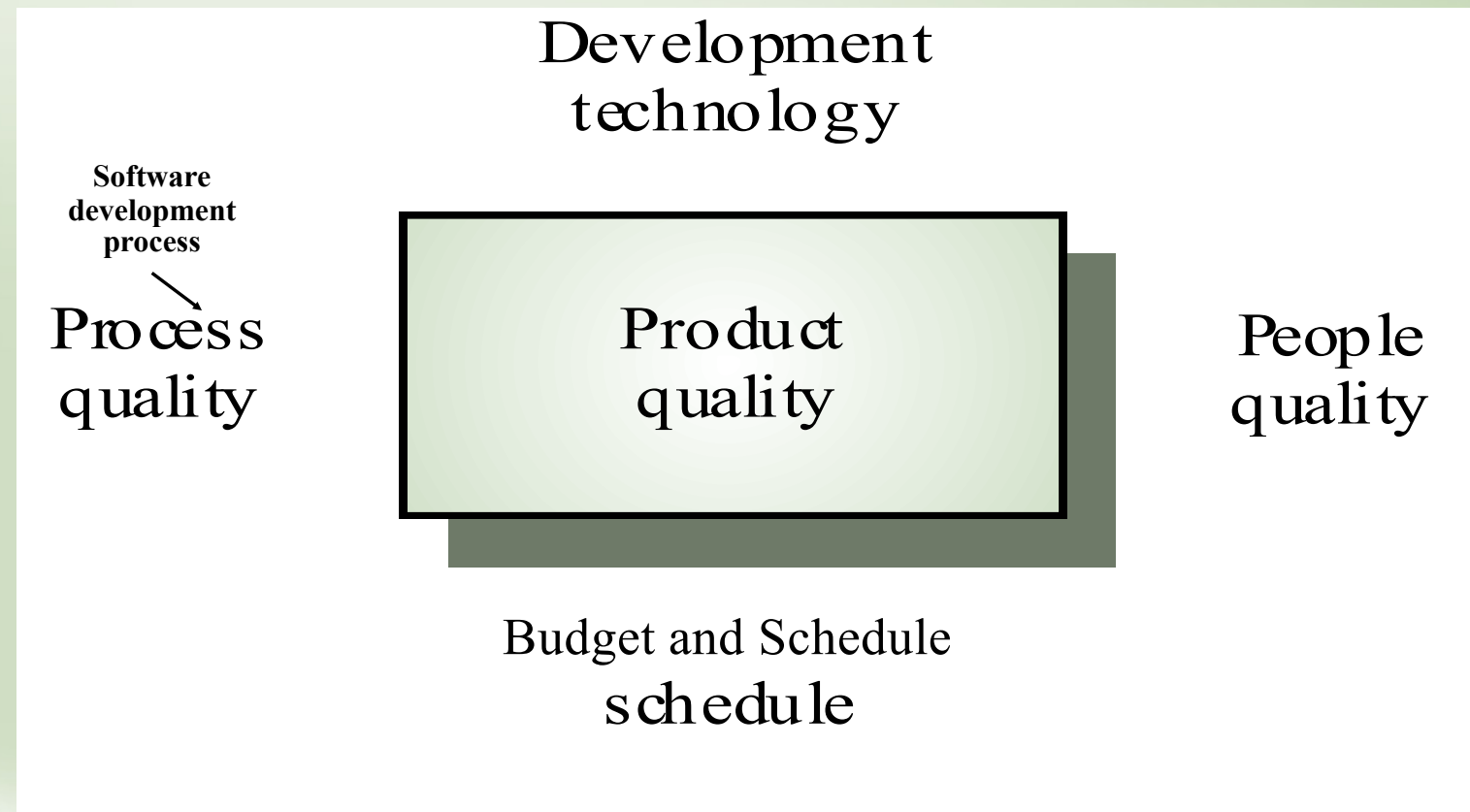
Quality attributes are frequently conflicting and increase development costs, so there is a need for weighting and balancing

Software engineering is concerned with the cost-effective development of good software

A product should meet its specification



Principal product quality factors (1)



Principal product quality factors (2)

Process quality

- ☐ A good process is usually required to produce a good product
- ☐ For manufactured goods, process is the principal quality determinant
- ☐ For design-based activity (like software development), other factors are also involved especially the capabilities of the designers
- ☐ For **large projects** with 'average' capabilities, the development process determines product quality

People quality

- ☐ For **small projects**, the capabilities of the developers is the main determinant
- ☐ Corollary: you need lower quality people (and higher quality process) in larger projects?



Principal product quality factors (3)

- **Development technology**
 - ❑ Is particularly significant for **small projects**
- **Budget and schedule**
 - ❑ Budget influences the Quality
 - ❑ In **all projects**, if an unrealistic schedule is imposed then product quality will suffer

Product quality attributes (1)

- | | |
|------------------------|---------------------|
| ❑ Efficiency | ❑ Robustness |
| ❑ Usability, | ❑ Understandability |
| ❑ Dependability | ❑ Testability |
| • reliability, | ❑ Adaptability |
| • availability, | ❑ Modularity |
| • security, | ❑ Simplicity |
| • safety | ❑ Portability |
| ❑ Maintainability | ❑ Reusability |
| ❑ Resilience | ❑ Learnability |



Product quality attributes (2)

☐ Attributes of good software (beyond delivering the required functionality):

☐ Efficiency

- ☐ Software should not make wasteful use of system resources (disk and memory space, CPU time, etc.) and should present appropriate response times

☐ Usability (ease of use)

- ☐ Software must be usable by the users for which it was designed

☐ Dependability (reliability, availability, security, safety,...)

- ☐ Software must be trustworthy

☐ Maintainability (ease of maintenance)

- ☐ Software must evolve to meet changing needs
- ☐ Software costs more to maintain than it does to develop. For systems with a long life, maintenance costs may be several times development costs

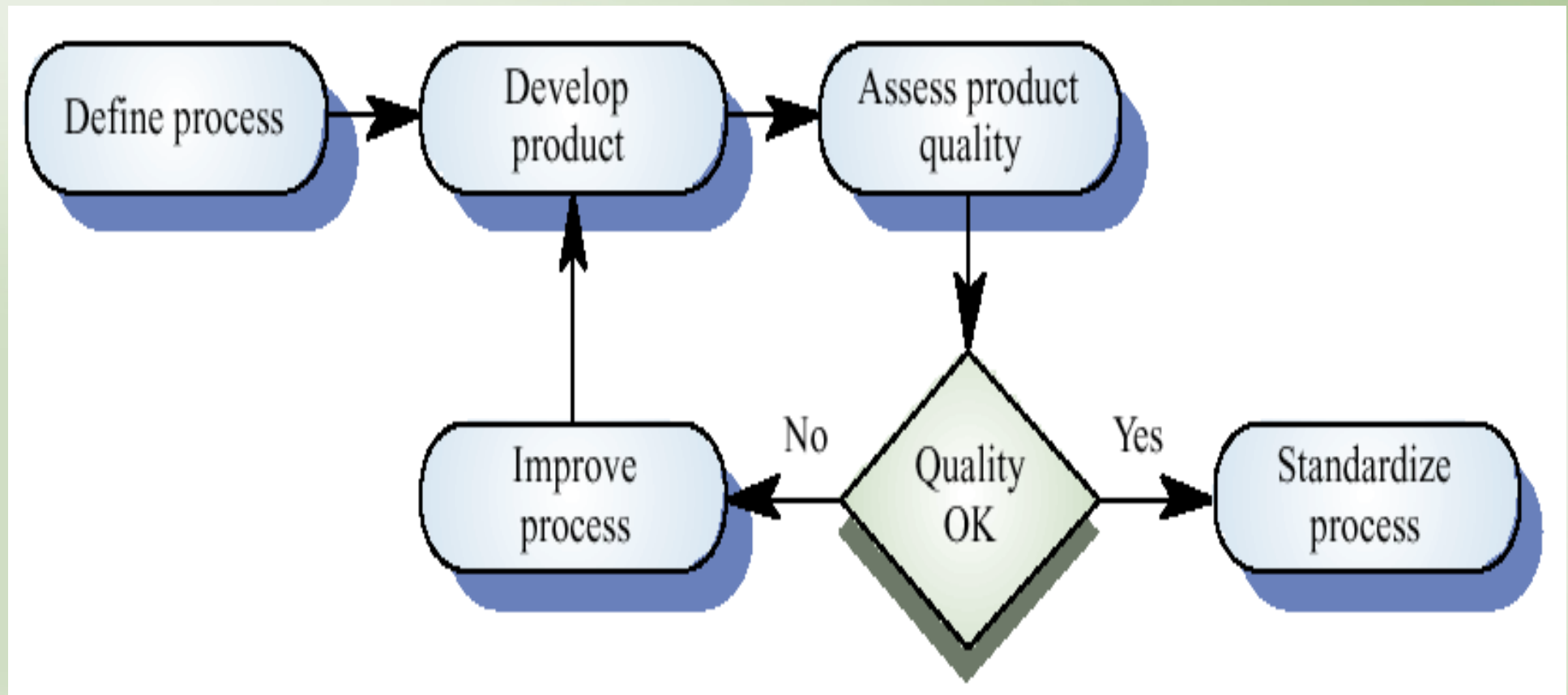


Process-based quality

- ❑ Straightforward link between process and product in manufactured goods
- ❑ More complex for software because:
 - The application of individual skills and experience is particularly important in software development
 - External factors such as the novelty of an application or the need for an accelerated development schedule may impair product quality
- ❑ Care must be taken not to impose inappropriate process standards



Process-based quality



Process quality attributes

Process characteristic	Description
Understandability	☐ To what extent is the process explicitly defined and how easy is it to understand the process definition?
Visibility	☐ Do the process activities culminate in clear results so that the progress of the process is externally visible?
Supportability	☐ To what extent can the process activities be supported by CASE tools?
Acceptability	☐ Is the defined process acceptable to and usable by the engineers responsible for producing the software product?



Process quality attributes

Process characteristic	Description
Reliability	☐ Is the process designed in such a way that process errors are avoided or trapped before they result in product errors?
Robustness	☐ Can the process continue in spite of unexpected problems?
Maintainability	☐ Can the process evolve to reflect changing organisational requirements or identified process improvements?
Rapidity	☐ How fast can the process of delivering a system from a given specification be completed?



Measuring Quality

- Measurement allows us to have a **quantitative view** of the quality concept.
 - Measurement allows us **to establish baselines** for qualities. Developers must know the **minimum level of quality** they must deliver for a product to be acceptable.
 - Organizations make continuous improvements in **their process models** and an improvement has a cost associated with it
 - Organizations need to know how much improvement in quality is achieved **at a certain cost** incurred due to process improvement.
 - The present level of quality of a product needs to be evaluated so the need for **improvements can be investigated**.
- ❑ **Measurement of User's View:** The user's view encompasses several quality factors, such as **functionality**, **reliability**, and **usability**. It is easy to measure how many of the functionalities a software product delivers by designing **at least one test case for each functionality**.

❑ **Measurement of Manufacturer's View:**

Defect Count: How many defects have been detected?

Rework Cost: How much does it cost to fix the known defects?

The **rework cost** can be split into two parts as follows:

Development Rework Cost: This is the rework cost incurred *before* a product is released to the customers. The development rework cost is a measure of development efficiency. In other words, if the development rework cost is zero, then the development efficiency is very high.

Operation Rework Cost: This is the rework cost incurred *when* a product is in operation. The operation rework cost is a measure of the delivered quality of the product in operation. If the development rework cost is zero, then the delivered quality of the product in operation is very high. This is because the customers have not encountered any defect and, consequently, the development team is not spending any resources on defect fixing



Quality factors and criteria

- **MCCALL'S QUALITY FACTORS** : A *quality factor* represents a **behavioural characteristic** of a system.
- Some examples of high-level quality factors are *correctness*, *reliability*, *efficiency*, *testability*, *portability*, and *reusability*.
- Quality factors are **external attributes** of a software system.
- **Customers**, **software developers**, and **quality assurance engineers** are interested in different quality factors to a different extent.

Eg: Customers may want **efficient** and **reliable** software with less concern for portability.

- The developers strive to meet **customer needs** by making their system efficient and reliable, at the same time making the **product portable** and **reusable** to reduce the cost of software development.
 - The software quality assurance team is more interested in the **testability** of a system so that some other factors, such as **correctness**, **reliability**, and **efficiency**, can be easily verified through testing.
- McCall had proposed **11 quality factors**



McCall's Quality Factors

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



Categorization of McCall's Quality Factors

❑ The **11 quality factors** have been grouped into three broad categories as follows:

1. **Product operation**

2. **Product revision**

3. **Product Transition**

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



Quality Criteria

❑ A *quality criterion* is an attribute of a quality factor that is **related to software development**.

For 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**.

Quality Criteria	Definition
Access audit	Ease with which software and data can be checked for compliance with standards or other requirements
Access control	Provisions for control and protection of the software and data
Accuracy	Precision of computations and output
Communication commonality	Degree to which standard protocols and interfaces are used
Completeness	Degree to which a full implementation of the required functionalities has been achieved



Quality Criteria

Quality Criteria	Definition
Communicativeness	Ease with which inputs and outputs can be assimilated
Conciseness	Compactness of the source code, in terms of lines of code
Consistency	Use of uniform design and implementation techniques and notation throughout a project
Data commonality	Use of standard data representations
Error tolerance	Degree to which continuity of operation is ensured under adverse conditions
Execution efficiency	Run time efficiency of the software
Expandability	Degree to which storage requirements or software functions can be expanded
Generality	Breadth of the potential application of software components
Hardware independence	Degree to which the software is dependent on the underlying hardware

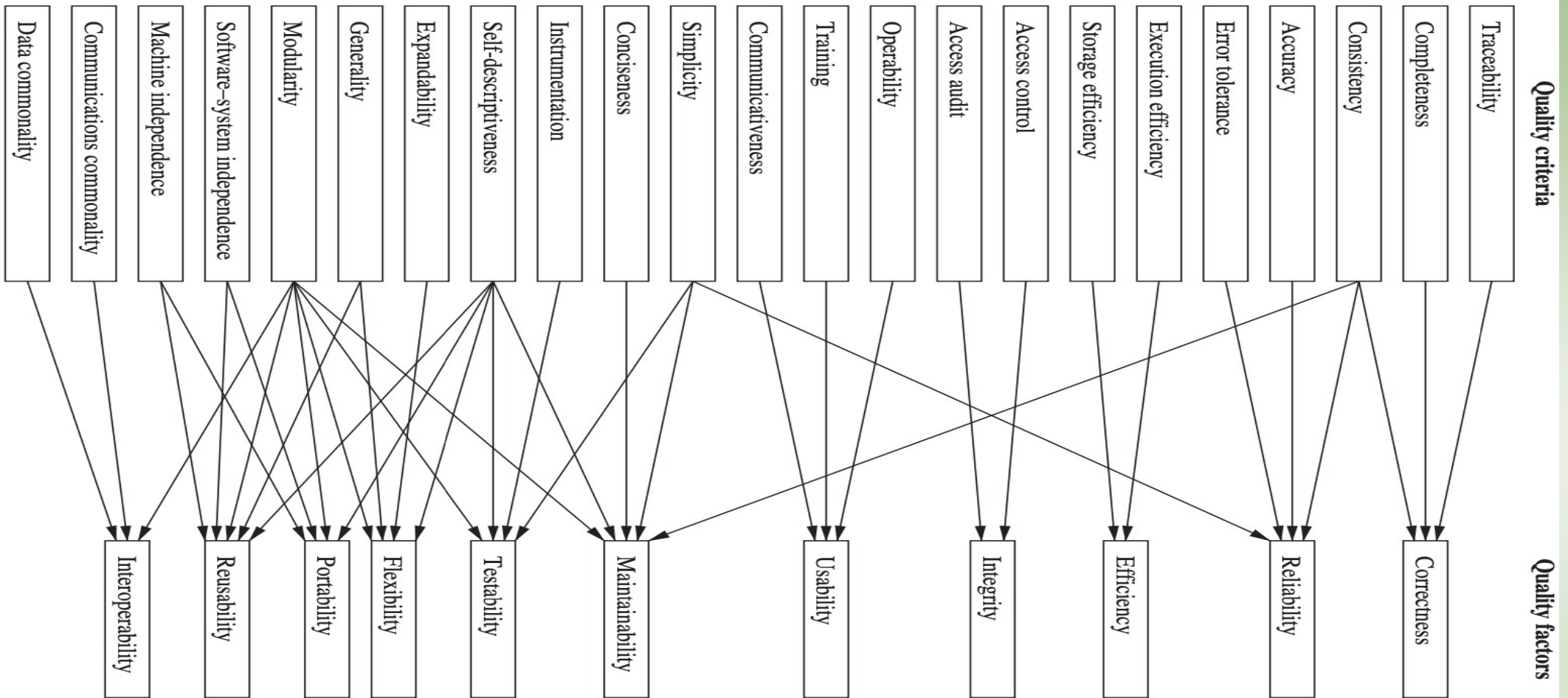


Quality Criteria

Quality Criteria	Definition
Instrumentation	Degree to which the software provides for measurement of its use or identification of errors
Modularity	Provision of highly independent modules
Operability	Ease of operation of the software
Self-documentation	Provision of in-line documentation that explains implementation of components
Simplicity	Ease with which the software can be understood.
Software system independence	Degree to which the software is independent of its software environment—nonstandard language constructs, operating system, libraries, database management system, etc.
Software efficiency	Run time storage requirements of the software
Traceability	Ability to link software components to requirements
Training	Ease with which new users can use the system



Relationship between quality factors and quality criteria



Quality Metrics

❑ The high-level quality factors **cannot be measured directly**.

For example, we can not directly measure the testability of a software system. Neither can testability be expressed in “yes” or “no” terms. Instead, the degree of testability can be assessed by associating with testability a few quality metrics, namely, *simplicity*, *instrumentation*, *self-descriptiveness*, and *modularity*.

❑ A *quality metric* is a measure that captures some aspect of a quality criterion. One or more quality metrics should be associated with each criterion. The metrics can be derived as follows:

1. Formulate a set of relevant questions concerning the quality criteria and seek a “yes” or “no” answer for each question.
2. Divide the number of “yes” answers by the number of questions to obtain a value in the range of 0 to 1. The resulting number represents the intended quality metric.



Quality Characteristics

- **ISO 9126 QUALITY CHARACTERISTICS:** There has been international collaboration among experts to define a general framework for software quality. An expert group, under the aegis of the ISO, standardized a software quality document, namely, ISO 9126, which defines six broad, independent categories of quality 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.
- **Reliability:** A set of attributes that bear on the capability of software to maintain its performance level under stated conditions for a stated period of time.
- **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.
- **Efficiency:** A set of attributes that bear on the relationship between the software's performance and the amount of resource used under stated conditions.
- **Maintainability:** A set of attributes that bear on the effort needed to make specified modifications (which may include corrections, improvements, or adaptations of software to environmental changes and changes in the requirements and functional specifications).
- **Portability:** A set of attributes that bear on the ability of software to be transferred from one environment to another (this includes the organizational, hardware or, software environment).



Quality Characteristics

ISO 9126 QUALITY SUB CHARACTERISTICS:

Suitability: The capability of the software to provide an adequate set of functions for specified tasks and user objectives.

Accuracy: The capability of the software to provide the right or agreed-upon results or effects.

Interoperability: The capability of the software to interact with one or more specified systems.

Security: The capability of the software to prevent unintended access and resist deliberate attacks intended to gain unauthorized access to confidential information or to make unauthorized modifications to information or to the program so as to provide the attacker with some advantage or so as to deny service to legitimate users.

Maturity: The capability of the software to avoid failure as a result of faults in the software.

Fault Tolerance: The capability of the software to maintain a specified level of performance in case of software faults or of infringement of its specified interface.

Recoverability: The capability of the software to reestablish its level of performance and recover the data directly affected in the case of a failure.

Understandability: The capability of the software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use.

Learnability: The capability of the software product to enable the user to learn its applications.

Operability: The capability of the software product to enable the user to operate and control it.

Attractiveness: The capability of the software product to be liked by the user.

Time Behavior: The capability of the software to provide appropriate response and processing times and throughput rates when performing its function under stated conditions.



Quality Characteristics

ISO 9126 QUALITY SUB CHARACTERISTICS:

Resource Utilization: The capability of the software to use appropriate resources in an appropriate time when the software performs its function under stated condition.

Analyzability: The capability of the software product to be diagnosed for deficiencies or causes of failures in the software or for the parts to be modified to be identified.

Changeability: The capability of the software product to enable a specified modification to be implemented.

Stability: The capability of the software to minimize unexpected effects from modifications of the software.

Testability: The capability of the software product to enable modified software to be validated.

Adaptability: The capability of the software to be modified for different specified environments without applying actions or means other than those provided for this purpose for the software considered.

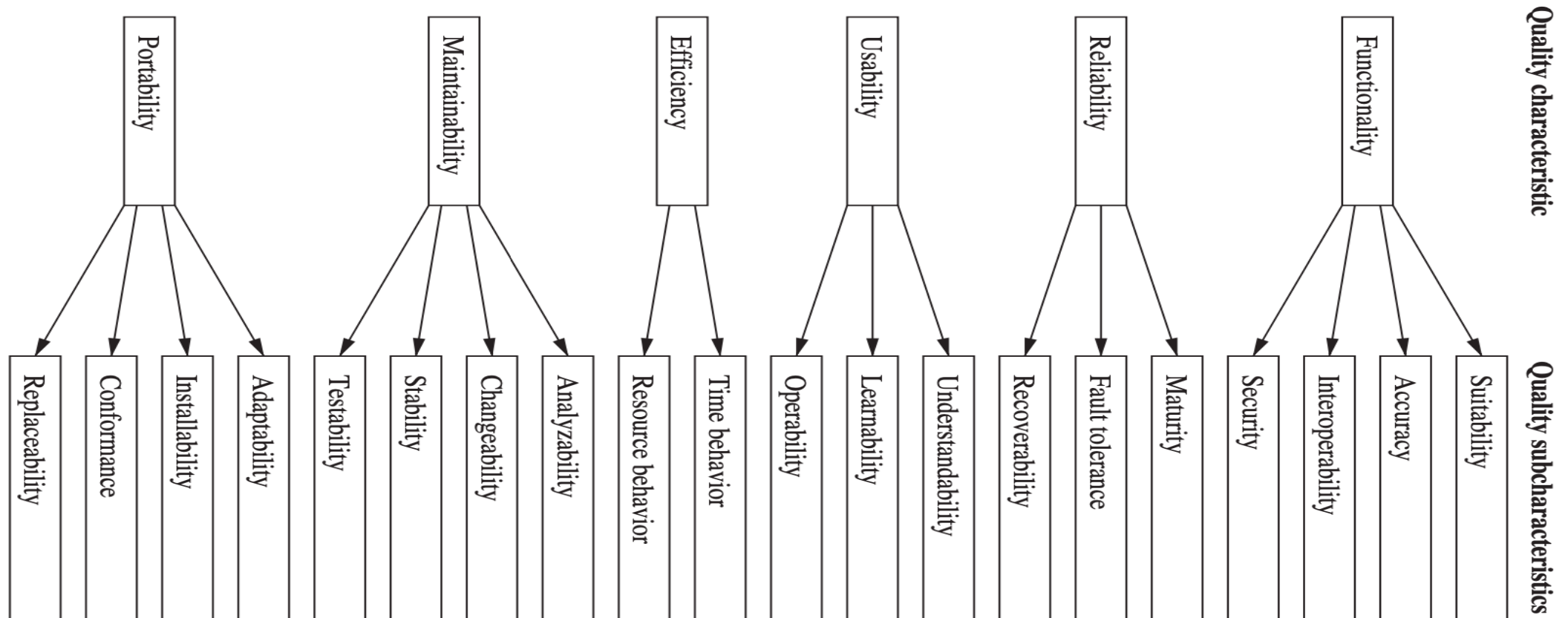
Installability: The capability of the software to be installed in a specified environment.

Coexistence: The capability of the software to coexist with other independent software in a common environment sharing common resources.

Replaceability: The capability of the software to be used in place of other specified software in the environment of that software.



ISO 9126 sample quality model refines the standard's features into sub characteristics



Differences between McCall and ISO9126 models

The following high-level **quality factors/characteristics** are found in both models: **reliability**, **usability**, **efficiency**, **maintainability**, and **portability**. However, there are several differences between the two models as explained in the following:

- ❑ The **ISO 9126 model** emphasizes **characteristics visible to the users**, whereas the McCall model considers **internal qualities** as well. For example, **reusability is an internal characteristic** of a product. Product developers strive to produce reusable components, whereas its impact is not perceived by customers.
- ❑ In McCall's model, one quality criterion can impact several quality factors, whereas in the ISO 9126 model, one sub-characteristic impacts exactly one quality characteristic.
- ❑ A high-level quality factor, such as testability, in the McCall model is a low-level sub-characteristic of maintainability in the ISO 9126 model.

Following are a few concerns with the quality models :

There is no consensus about what high-level quality factors are most important at the top level. McCall et al. suggest **11 high-level quality factors**, whereas the ISO 9126 standard defines only **6 quality characteristics**. Some of the quality factors in the McCall model are more important to **developers**. For example, reusability and interoperability are important to developers. However, the ISO 9126 model just considers the product.

There is no consensus regarding what is a top-level quality factor/ characteristic and what is a more concrete quality criterion/subcharacteristic. These days many applications run on computer and communications networks. However, interoperability is not an independent, top-level quality characteristic in the ISO 9126 model. It is not clear why interoperability is a part of functionality. The absence of a rationale makes it difficult to follow a prescribed quality model.



Software Quality Standards

- The ISO has developed a series of standards, collectively known as the **ISO 9000**.
- The ISO was **founded in 1946**, and it is based in **Geneva, Switzerland**.
- It develops and promotes **international standards** in the field of quality assurance and quality management.
- The ISO 9000 standards are generally applicable to all tangible products manufactured with human endeavour, say, from spices to software
- The ISO 9000 standards are reviewed and updated from time to time, **once every 5–8 years**.
- The latest ISO 9000 standards released in the year 2000 are referred to as **ISO 9000:2000**
- There are three components of the ISO 9000:2000 standard as follows:
 - **ISO 9000 : Fundamentals and vocabulary**
 - ISO 9001 : Requirements**
 - ISO 9004 : Guidelines for performance improvements**
- At this point we remind the reader that ISO 9002 and ISO 9003 were parts of ISO 9000:1994, but these are no longer parts of ISO 9000:2000. ISO 9002 dealt with the quality system model for quality assurance in production and installation, whereas ISO 9003 dealt with the quality system model for quality assurance in final inspection and testing.
- **ISO 9000:2000 Fundamentals**
 - The ISO 9000:2000 standard is based on the following eight principles: **Principle 1. Customer Focus: Principle 2. Leadership, Principle 3. Involvement of People, Principle 4. Process Approach, Principle 5. System Approach to Management, Principle 6. Continual Improvement, Principle 7. Factual Approach to Decision Making, Principle 8. Mutually Beneficial Supplier Relationships**



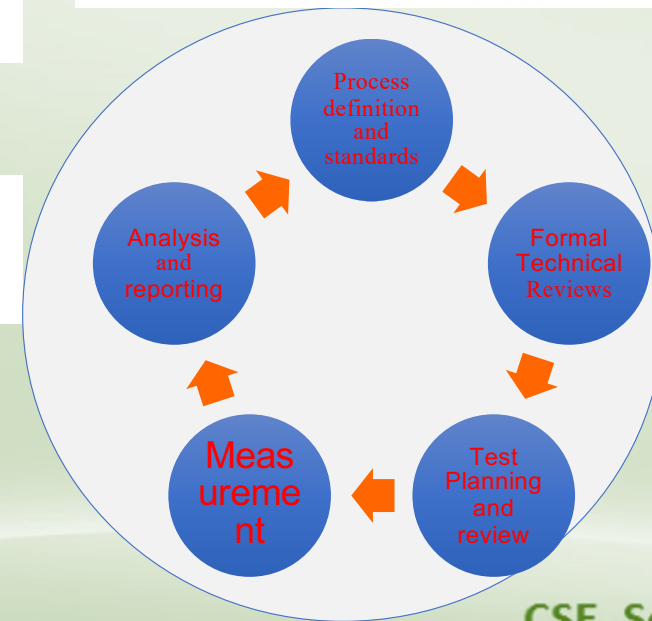
Software Quality Assurance Group

The **SQA group** is a team of people with the necessary training and skills to ensure that

- ☐ All necessary actions are taken during the development process.
- ☐ So that the **resulting software conforms to established technical requirements.**



SQA group must look at software from the customer's **perspective assessing its technical merits.**



SQA Group Activities

- ☐ Prepare **SQA plan** for the project.
- ☐ Participate in the **development of the project's software process Description**.
- ☐ Review **software engineering activities to verify compliance with the defined software process**.
- ☐ **Audit** designated software work products to verify compliance with those **defined** as part of the software process.
- ☐ **Ensure that any deviations** in software or work products are documented and handled according to a documented procedure.
- ☐ **Record any evidence of noncompliance** and reports them to management.



IEEE Standards for SQA Plan

- ❑ An important part of achieving quality is **to plan for quality**.
- ❑ A Software Quality Assurance Plan is **NOT** merely another name for **a test plan**, though test plans are included in an SQA plan.
- ❑ The IEEE Standards Association has developed a standard (Std 730-1989) for software quality assurance plans.

The following is part of the sections specified in IEEE Std 730-1989

Purpose: This section shall list the software covered and the portions of software life cycle covered.

Reference Documents: This section shall list all the documents referenced in the plan.



IEEE Standards for SQA Plan

3. Management

3.1. Organization: This selection describes the structure of organization and the responsibilities, and usually includes an organizational chart.

3.2. Tasks: This section lists all of the tasks to be performed, relationship between tasks and checkpoints, and the sequence of the tasks.

3.3. Responsibilities: This section shall list the responsibilities of each organizational unit.

4. Documentation

4.1. Purpose: required document and state how documents will be evaluated.

4.2. Minimum documents: This section shall describe the minimum required documentation, usually including the following:

SRs—software Requirements Specification

SDD—Software Design Description

SVVP—Software Verification and Validation Plan

SVVR—Software Verification and Validation Report

User documentation & **SCMP**—SW Configuration Management Plan



IEEE Standards for SQA Plan

5. Standards, Practices, Conventions, and Metrics

- ☐ This section shall identify the S, P, C, and M to be applied and how compliance is to be monitored and assured.
- ☐ The minimal contents include documentation standards, logic structure standards, coding standards, testing standards, selected SQA product, and process metrics.

6. Reviews and Audits: This section shall define what reviews/audits will be done, how they will be accomplished, and what further actions are required.

7. Tests: This section shall include all tests that are not included in SVVP.
[software verification and Validation Plan]

8. Problem Reporting: This section shall define practices and procedures for reporting, tracking, and resolving problems, including organizational responsibilities.



IEEE Standards for SQA Plan

- 9. **Tools, Techniques, and Methodologies:** This section shall identify the special software tools, techniques, and methodologies and describe their use.
- 10. **Code Control:** This section shall define the methods and facilities to maintain controlled versions of the software.
- 11. **Media Control:** This section shall define the methods and facilities to identify, store, and protect the physical media.
- 12. **Supplier Control (for outsourcing):** This section shall state provisions for assuring that software provided by suppliers meets standards.
- 13. **Records:** This section shall identify documentation to be retained and methods to collection, maintain, and safeguard the documentation.
- 14. **Training:** This section shall identify necessary training activities.
- 15. **Risk Management:** This section shall specify methods and procedures for risk management.



Quality Standards

Process:

ISO 9000 for process mapping
SEI CMM for software process improvement
SEI CMMi for integrated software process

People:

PCMM for people involvement in quality initiatives (Competencies & Talent)

People & Process (mix and match)

Six-Sigma initiative for continuous improvement on existing processes

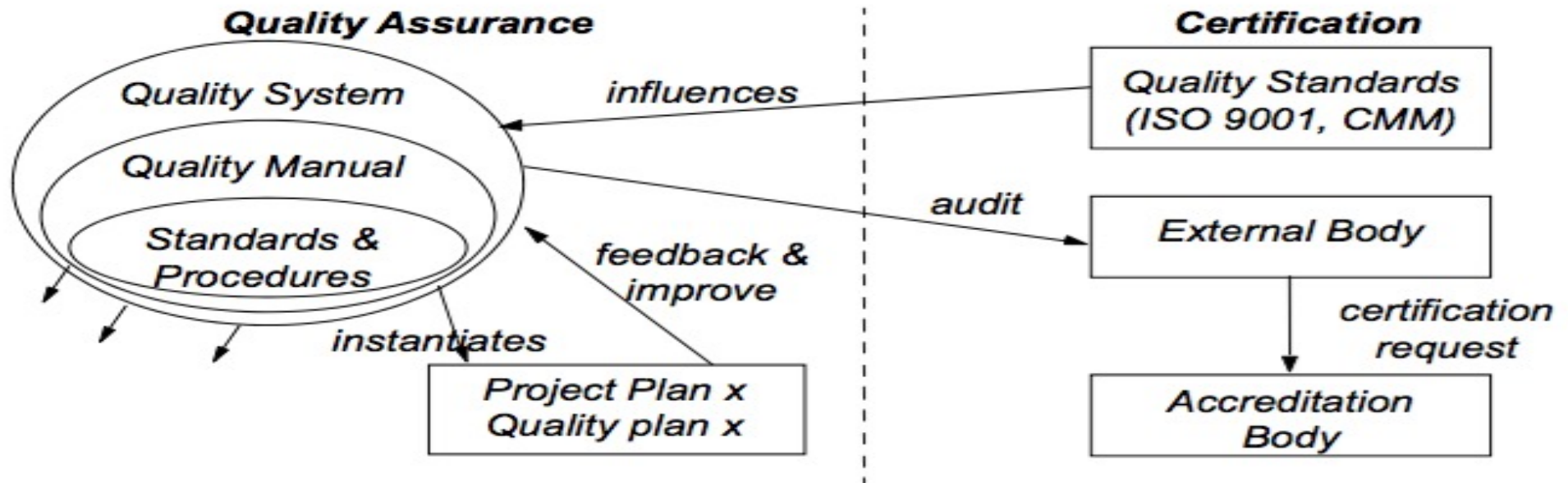
Other Quality Methodologies

Bootstrap
TRILLIUM
Quality Improvement Paradigm
Kaizen



Quality System

A Quality Plan should be an **instance** of an organization's Quality System.



Customers may require an externally reviewed quality system



ISO 9000

ISO 9000 is an international **set of standards** for **quality management** applicable to a range of organizations from manufacturing to service industries.

ISO 9001 is a generic model for **quality process**, applicable to organizations whose business processes range all the way **from design and development, to production, installation and servicing**;

- **ISO 9001 must be instantiated for each organization**
- **ISO 90003** interpret: **ISO 9001 for the software developer**

ISO = International Organization for Standardization

- ISO main site: <http://www.iso.ch/>
- ISO 9000 main site: <http://www.tc176.org/>



Capability Maturity Model (CMM)

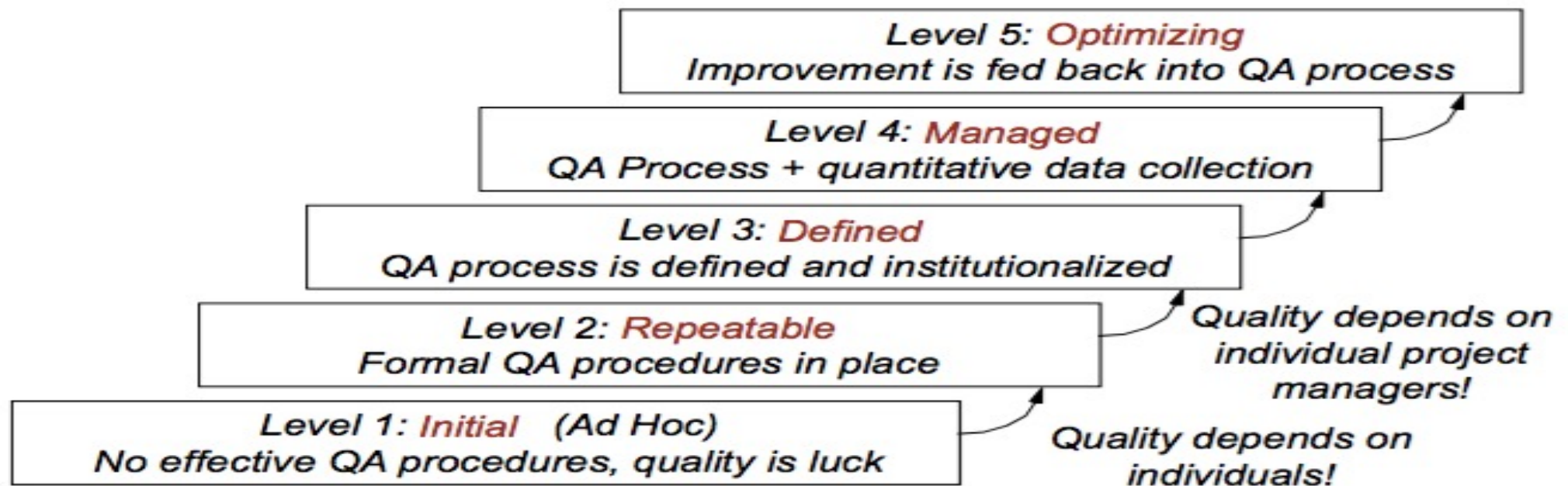
The SEI process maturity model classifies

- ❑ how well contractors manage software processes

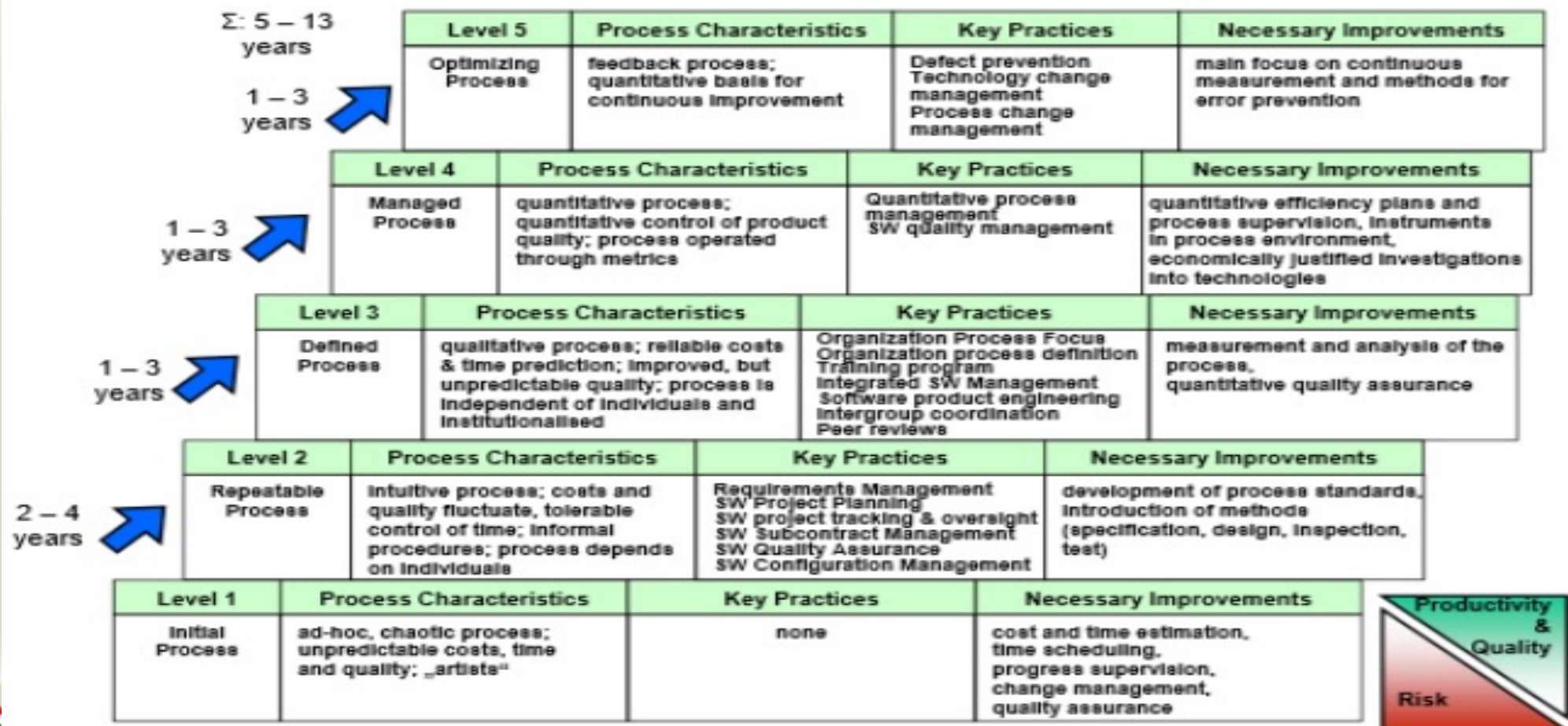
CMM Links -

<http://www.sei.cmu.edu/cmm/cmm.sum.html>

<http://www2.umassd.edu/SWPI/processframework/cmm/cmm.html>



CMMI



PCMM :

- ❑ **Strong emphasis on learning and professional development**, both at the individual and organizational levels
- ❑ Culture that **empowers individuals** and promotes a participatory environment
- ❑ Effective professional **mentoring** programs
- ❑ Emphasis **on continuous improvement**

Six-Sigma:

- ❑ It is a customer based approach realizing that **defects are expensive**.
- ❑ It is a **structured application of tools & techniques** applied on the project basis to achieve sustainable results
- ❑ **Six Sigma processes** will produce **less than 3.4 defects /mistakes** for a million opportunities i.e. 99.99% ,perfection [Called five 9's]



QUESTIONS PLEASE 😊

