Jo kaam karna hai
vo hora ki nahi } → sabse jyda jarroori

UNIT 4

Software Reliability means **Operational reliability**. It is described as the ability of a system or component to perform its required functions under static conditions for a specific period.

Software reliability is also defined as the probability that a software system fulfills its assigned task in a given environment for a predefined number of input cases, assuming that the hardware and the input are free of error.

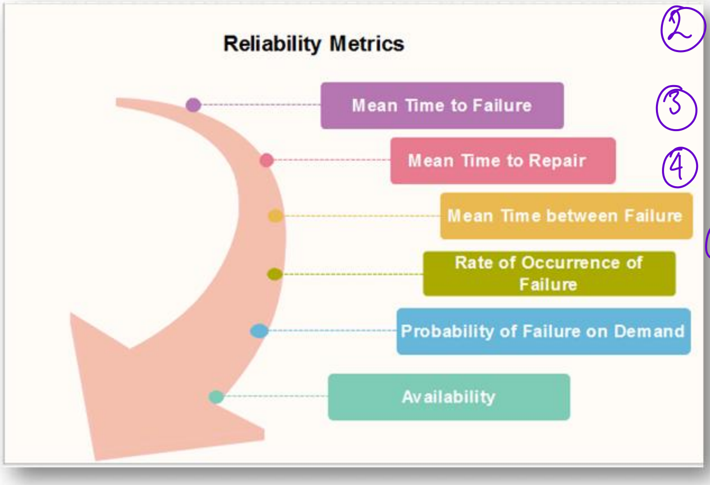
Software Reliability is an essential connect of software quality, composed with functionality, usability, performance, serviceability, capability, installability, maintainability, and documentation. Software Reliability is hard to achieve because the complexity of software turn to be high. While any system with a high degree of complexity, containing software, will be hard to reach a certain level of reliability, system developers tend to push complexity into the software layer, with the speedy growth of system size and ease of doing so by upgrading the software.

For example, large next-generation aircraft will have over 1 million source lines of software on-board; next-generation air traffic control systems will contain between one and two million lines; the upcoming International Space Station will have over two million lines on-board and over 10 million lines of ground support software; several significant life-critical defense systems will have over 5 million source lines of software. While the complexity of software is inversely associated with software reliability, it is directly related to other vital factors in software quality, especially functionality, capability, etc.

Reliability Metrics

Reliability metrics are used to quantitatively expressed the reliability of the software product. The option of which metric is to be used depends upon the type of system to which it applies & the requirements of the application domain.

Some reliability metrics which can be used to quantify the reliability of the software product are as follows:


Reliability Metrics
Mean Time to Failure
Mean Time to Repair
Mean Time between Failure
Rate of Occurrence of Failure
Probability of Failure on Demand
Availability

① Mean Time to failure (MTTF) $\sum_{i=1}^{n}(t_{i+1} - t_i)/n-1$

② Mean Time to Repair (MTTR) → Thik karne ka time

③ Mean Time Between Failure ⇒ MTTF+ MTTR

④ Probablity of Failure on Demand → Kaam k time fakchodi

⑤ Rate of Occurrence of failure # failure / time unit

⑥ Availablity → Khali kitne time hai ?

1. Mean Time to Failure (MTTF)

MTTF is described as the time interval between the two successive failures. An **MTTF** of 200 mean that one failure can be expected each 200-time units. The time units are entirely dependent on the system & it can even be stated in the number of transactions. **MTTF** is consistent for systems with large transactions.

For example, It is suitable for computer-aided design systems where a designer will work on a design for several hours as well as for Word-processor systems.

To measure **MTTF**, we can evidence the failure data for n failures. Let the failures appear at the time instants $\mathbf{t_1,t_2.....t_n}$.

MTTF can be calculated as

$$\sum_{i=1}^{n} \frac{t_{i+1}-t_i}{(n-1)}$$

2. Mean Time to Repair (MTTR)

Once failure occurs, some-time is required to fix the error. **MTTR** measures the average time it takes to track the errors causing the failure and to fix them.

3. Mean Time Between Failure (MTBR)

We can merge **MTTF & MTTR** metrics to get the MTBF metric.

MTBF = MTTF + MTTR

Thus, an **MTBF** of 300 denoted that once the failure appears, the next failure is expected to appear only after 300 hours. In this method, the time measurements are real-time & not the execution time as in **MTTF**.

4. Rate of occurrence of failure (ROCOF)

It is the number of failures appearing in a unit time interval. The number of unexpected events over a specific time of operation. **ROCOF** is the frequency of occurrence with which unexpected role is likely to appear. A **ROCOF** of 0.02 mean that two failures are likely to occur in each 100 operational time unit steps. It is also called the failure intensity metric.

5. Probability of Failure on Demand (POFOD)

POFOD is described as the probability that the system will fail when a service is requested. It is the number of system deficiency given several systems inputs.

POFOD is the possibility that the system will fail when a service request is made.

A **POFOD** of 0.1 means that one out of ten service requests may fail.**POFOD** is an essential measure for safety-critical systems. POFOD is relevant for protection systems where services are demanded occasionally.

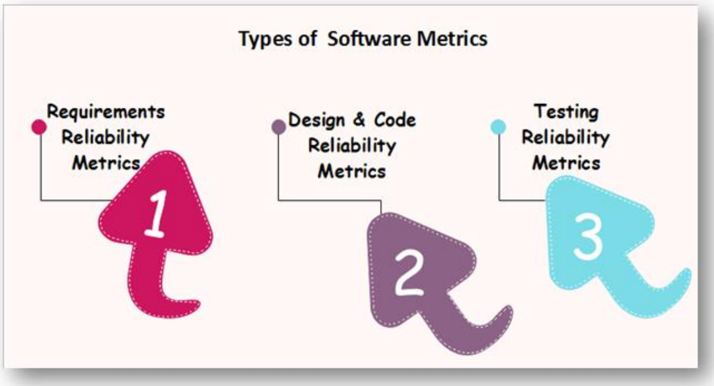
6. Availability (AVAIL)

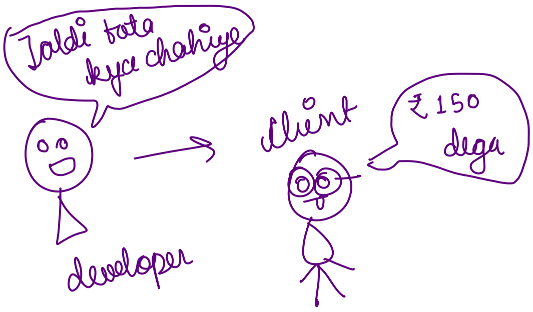
Availability is the probability that the system is applicable for use at a given time. It takes into account the repair time & the restart time for the system. An availability of 0.995 means that in every 1000 time units, the system is feasible to be available for **995** of these. The percentage of time that a system is applicable for use, taking into account planned and unplanned downtime. If a system is down an average of four hours out of 100 hours of operation, its **AVAIL** is 96%.

Software Metrics for Reliability

The Metrics are used to improve the reliability of the system by identifying the areas of requirements.

Different Types of Software Metrics are:-


Types of Software Metrics
Requirements Reliability Metrics
1
Design & Code Reliability Metrics
2
Testing Reliability Metrics
3



Requirements Reliability Metrics

{ kya kya chahiye }

Requirements denote what features the software must include. It specifies the functionality that must be contained in the software. The requirements must be written such that is no misconception between the developer & the client. The requirements must include valid structure to avoid the loss of valuable data.

very serious, no bakchodi here

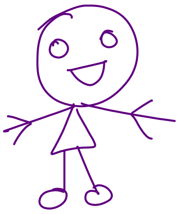
The requirements should be thorough and in a detailed manner so that it is simple for the design stage. The requirements should not include inadequate data. Requirement Reliability metrics calculates the above-said quality factors of the required document.

Design and Code Reliability Metrics

The quality methods that exists in design and coding plan are complexity, size, and modularity. Complex modules are tough to understand & there is a high probability of occurring bugs. The reliability will reduce if modules have a combination of high complexity and large size or high complexity and small size. These metrics are also available to object-oriented code, but in this, additional metrics are required to evaluate the quality.

Testing Reliability Metrics

① Jhatu system nahi hona chahiye
② good test plan + Quality test cases



These metrics use two methods to calculate reliability.

First, it provides that the system is equipped with the tasks that are specified in the requirements. Because of this, the bugs due to the lack of functionality reduces.

The **second** method is calculating the code, finding the bugs & fixing them. To ensure that the system includes the functionality specified, test plans are written that include multiple test cases. Each test method is based on one system state and tests some tasks that are based on an associated set of requirements. The goals of an effective verification program is to ensure that each elements is tested, the implication being that if the system passes the test, the requirements functionality is contained in the delivered system.

Software Measurement: A measurement is a manifestation of the size, quantity, amount, or dimension of a particular attribute of a product or process. Software measurement is a titrate impute of a characteristic of a software product or the software process. It is an authority within software engineering. The software measurement process is defined and governed by ISO Standard.

Software Measurement Principles:

Formulation
Collection
Analysis
Interpretation
Feedback

The software measurement process can be characterized by five activities-

1. **Formulation:** The derivation of software measures and metrics appropriate for the representation of the software that is being considered.
2. **Collection:** The mechanism used to accumulate data required to derive the formulated metrics.
3. **Analysis:** The computation of metrics and the application of mathematical tools.
4. **Interpretation:** The evaluation of metrics resulting in insight into the quality of the representation.
5. **Feedback:** Recommendation derived from the interpretation of product metrics transmitted to the software team.

Need for Software Measurement:

Software is measured to:

- Create the quality of the current product or process.
- Anticipate future qualities of the product or process.
- Enhance the quality of a product or process.
- Regulate the state of the project in relation to budget and schedule.

Classification of Software Measurement:

Direct
Indirect

There are 2 types of software measurement:

1. **Direct Measurement:** In direct measurement, the product, process, or thing is measured directly using a standard scale.

2. **Indirect Measurement:** In indirect measurement, the quantity or quality to be measured is measured using related parameters i.e. by use of reference.

Metrics:

A metric is a measurement of the level at which any impute belongs to a system product or process.

Software metrics will be useful only if they are characterized effectively and validated so that their worth is proven. There are 4 functions related to software metrics:

1. Planning
2. Organizing
3. Controlling
4. Improving

Characteristics of software Metrics:

numerical values

1. **Quantitative:** Metrics must possess quantitative nature. It means metrics can be expressed in values. samaj m aaya kya
2. **Understandable:** Metric computation should be easily understood, and the method of computing metrics should be clearly defined.
3. **Applicability:** Metrics should be applicable in the initial phases of the development of the software. sab jagah lag jaye
4. Baar Baar **Repeatable:** The metric values should be the same when measured repeatedly and consistent in nature.
5. **Economical:** The computation of metrics should be economical.
6. **Language Independent:** Metrics should not depend on any programming language.

Classification of Software Metrics:

There are 3 types of software metrics:

1. **Product Metrics:** Product metrics are used to evaluate the state of the product, tracing risks and undercover prospective problem areas. The ability of the team to control quality is evaluated.
2. **Process Metrics:** Process metrics pay particular attention to enhancing the long-term process of the team or organization.
3. **Project Metrics:** The project matrix describes the project characteristic and execution process.
    - Number of software developer
    - Staffing patterns over the life cycle of software
    - Cost and schedule
    - Productivity

Advantages of Software Metrics :

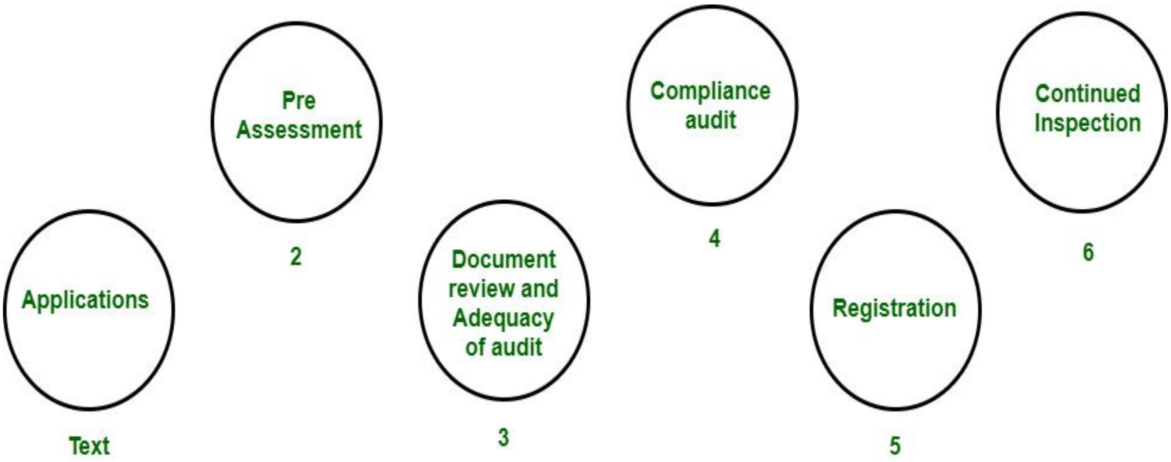
1. Reduction in cost or budget.
2. It helps to identify the particular area for improvising.

3. It helps to increase the product quality.
4. Managing the workloads and teams.
5. Reduction in overall time to produce the product,.
6. It helps to determine the complexity of the code and to test the code with resources.
7. It helps in providing effective planning, controlling and managing of the entire product.

Disadvantages of Software Metrics :

1. It is expensive and difficult to implement the metrics in some cases.
2. Performance of the entire team or an individual from the team can't be determined. Only the performance of the product is determined.
3. Sometimes the quality of the product is not met with the expectation.
4. It leads to measure the unwanted data which is wastage of time.
5. Measuring the incorrect data leads to make wrong decision making.

ISO: The International organization for Standardization is a world wide federation of national standard bodies. The **International standards organization (ISO)** is a standard which serves as a for contract between independent parties. It specifies guidelines for development of **quality system**. Quality system of an organization means the various activities related to its products or services. Standard of ISO addresses to both aspects i.e. operational and organizational aspects which includes responsibilities, reporting etc. An ISO 9000 standard contains set of guidelines of production process without considering product itself.



ISO 9000 Certification

Why ISO Certification required by Software Industry?
There are several reasons why software industry must get an ISO certification. Some of reasons are as follows :

- This certification has become a standards for international bidding.
- It helps in designing high-quality repeatable software products.
- It emphasis need for proper documentation.
- It facilitates development of optimal processes and totally quality measurements.

Features of ISO 9001 Requirements :

- **Document control –**
  All documents concerned with the development of a software product should be properly managed and controlled.
- **Planning –**
  Proper plans should be prepared and monitored.

- **Review –**
  For effectiveness and correctness all important documents across all phases should be independently checked and reviewed .
- **Testing –**
  The product should be tested against specification.
- **Organizational Aspects –**
  Various organizational aspects should be addressed e.g., management reporting of the quality team.

Advantages of ISO 9000 Certification :

Some of the advantages of the ISO 9000 certification process are following :

- Business ISO-9000 certification forces a corporation to specialize in "how they are doing business". Each procedure and work instruction must be documented and thus becomes a springboard for continuous improvement.
- Employees morale is increased as they're asked to require control of their processes and document their work processes
- Better products and services result from continuous improvement process.
- Increased employee participation, involvement, awareness and systematic employee training are reduced problems.

Shortcomings of ISO 9000 Certification :

Some of the shortcoming of the ISO 9000 certification process are following :

- ISO 9000 does not give any guideline for defining an appropriate process and does not give guarantee for high quality process.
- ISO 9000 certification process have no international accreditation agency exists.

CMM was developed by the Software Engineering Institute (SEI) at Carnegie Mellon University in 1987.

- It is not a software process model. It is a framework that is used to analyze the approach and techniques followed by any organization to develop software products.
- It also provides guidelines to further enhance the maturity of the process used to develop those software products.
- It is based on profound feedback and development practices adopted by the most successful organizations worldwide.
- This model describes a strategy for software process improvement that should be followed by moving through 5 different levels.
- Each level of maturity shows a process capability level. All the levels except level-1 are further described by Key Process Areas (KPA's).

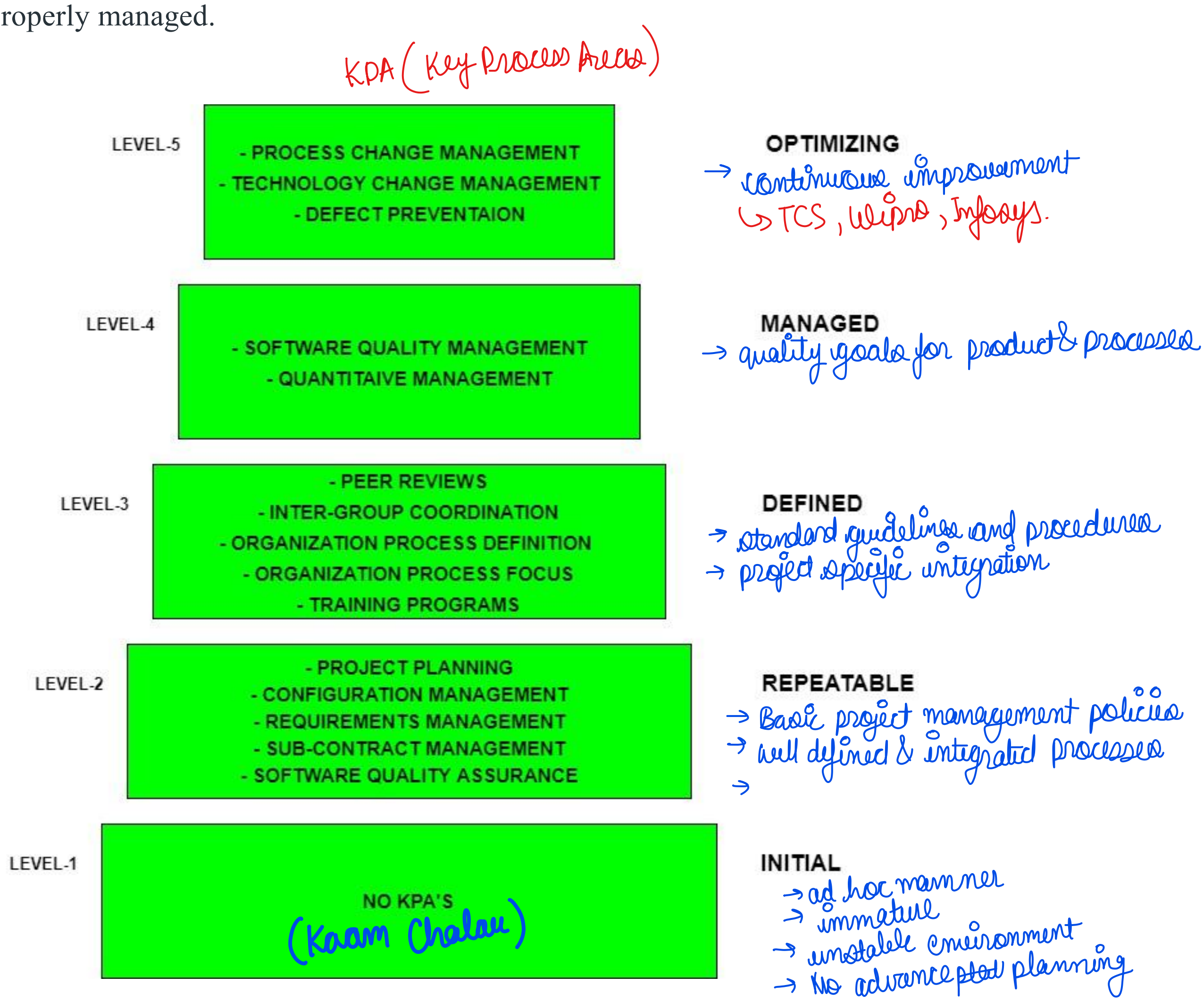
Shortcomings of SEI/CMM:

- It encourages the achievement of a higher maturity level in some cases by displacing the true mission, which is improving the process and overall software quality.
- It only helps if it is put into place early in the software development process.
- It has no formal theoretical basis and in fact is based on the experience of very knowledgeable people.
- It does not have good empirical support and this same empirical support could also be constructed to support other models.

Key Process Areas (KPA's):

Each of these KPA's defines the basic requirements that should be met by a software process in order to satisfy the KPA and achieve that level of maturity.

Conceptually, key process areas form the basis for management control of the software project and establish a context in which technical methods are applied, work products like models, documents, data, reports, etc. are produced, milestones are established, quality is ensured and change is properly managed.


KPA (Key Process Areas)
LEVEL-5
- PROCESS CHANGE MANAGEMENT
- TECHNOLOGY CHANGE MANAGEMENT
- DEFECT PREVENTAION
OPTIMIZING
→ continuous improvement
↳ TCS, Wipro, Infosys.
LEVEL-4
- SOFTWARE QUALITY MANAGEMENT
- QUANTITAIVE MANAGEMENT
MANAGED
→ quality goals for product & processes
LEVEL-3
- PEER REVIEWS
- INTER-GROUP COORDINATION
- ORGANIZATION PROCESS DEFINITION
- ORGANIZATION PROCESS FOCUS
- TRAINING PROGRAMS
DEFINED
→ standard guidelines and procedures
→ project specific integration
LEVEL-2
- PROJECT PLANNING
- CONFIGURATION MANAGEMENT
- REQUIREMENTS MANAGEMENT
- SUB-CONTRACT MANAGEMENT
- SOFTWARE QUALITY ASSURANCE
REPEATABLE
→ Basic project management policies
→ well defined & integrated processes
→
LEVEL-1
NO KPA'S
(Kaam Chalau)
INITIAL
→ ad hoc manner
→ immature
→ unstable environment
→ No advance planning

The 5 levels of CMM are as follows:

Level-1: Initial –

- No KPA's defined.
- Processes followed are Adhoc and immature and are not well defined.
- Unstable environment for software development.
- No basis for predicting product quality, time for completion, etc.

Level-2: Repeatable –

- Focuses on establishing basic project management policies.
- Experience with earlier projects is used for managing new similar natured projects.
- **Project Planning-** It includes defining resources required, goals, constraints, etc. for the project. It presents a detailed plan to be followed systematically for the successful completion of good quality software.
- **Configuration Management-** The focus is on maintaining the performance of the software product, including all its components, for the entire lifecycle.
- **Requirements Management-** It includes the management of customer reviews and feedback which result in some changes in the requirement set. It also consists of accommodation of those modified requirements.
- **Subcontract Management-** It focuses on the effective management of qualified software contractors i.e. it manages the parts of the software which are developed by third parties.

- **Software Quality Assurance-** It guarantees a good quality software product by following certain rules and quality standard guidelines while developing.

Level-3: Defined –

- At this level, documentation of the standard guidelines and procedures takes place.
- It is a well-defined integrated set of project-specific software engineering and management processes.
- **Peer Reviews-** In this method, defects are removed by using a number of review methods like walkthroughs, inspections, buddy checks, etc.
- **Intergroup Coordination-** It consists of planned interactions between different development teams to ensure efficient and proper fulfillment of customer needs.
- **Organization Process Definition-** Its key focus is on the development and maintenance of the standard development processes.
- **Organization Process Focus-** It includes activities and practices that should be followed to improve the process capabilities of an organization.
- **Training Programs-** It focuses on the enhancement of knowledge and skills of the team members including the developers and ensuring an increase in work efficiency.

Level-4: Managed –

- At this stage, quantitative quality goals are set for the organization for software products as well as software processes.
- The measurements made help the organization to predict the product and process quality within some limits defined quantitatively.
- **Software Quality Management-** It includes the establishment of plans and strategies to develop quantitative analysis and understanding of the product's quality.
- **Quantitative Management-** It focuses on controlling the project performance in a quantitative manner.

Level-5: Optimizing –

- This is the highest level of process maturity in CMM and focuses on continuous process improvement in the organization using quantitative feedback.
- Use of new tools, techniques, and evaluation of software processes is done to prevent recurrence of known defects.
- **Process Change Management-** Its focus is on the continuous improvement of the organization's software processes to improve productivity, quality, and cycle time for the software product.
- **Technology Change Management-** It consists of the identification and use of new technologies to improve product quality and decrease product development time.
- **Defect Prevention-** It focuses on the identification of causes of defects and prevents them from recurring in future projects by improving project-defined processes.

Comparison between ISO and SEI CMM

ISO 9000:

It is a set of International Standards on quality management and quality assurance developed to help companies effectively document the quality system elements needed to an efficient quality system.

SEICMM:

SEI (Software Engineering Institute), Capability Maturity Model (CMM) specifies an increasing series of levels of a software development organization.

Difference between ISO9000 and SEI-CMM:

ISO 9000	SEICMM
ISO 9000 is a set of international standards on quality management and quality assurance developed to help companies effectively document the quality system elements needed to an efficient quality system. (process and product)	SEI (Software Engineering Institute), Capability Maturity Model (CMM) specifies an increasing series of levels of a software development organization. (organization)
→ customer ka katna nahi chahiye Focus is customer supplier relationship, attempting to reduce customer's risk in choosing a supplier.	Focus on the software supplier to improve its interval processes to achieve a higher quality product for the benefit of the customer. → Kaam acha karo
It is created for hard goods manufacturing industries.	It is created for software industry. (Specifically)
ISO9000 is recognized and accepted in most of the countries. ↑ Jalwa hai hamara har jagah	SEICMM is used in USA, less widely elsewhere.
It specifies concepts, principles and safeguards that should be in place.	CMM provides detailed and specific definition of what is required for given levels.
This establishes one acceptance level.	It assesses on 5 levels.
Its certification is valid for three years.	It has no limit on certification.
It focuses on inwardly processes.	It focus outwardly.
It has no level.	It has 5 levels: **(a).** Initial **(b).** Repeatable **(c).** Defined **(d).** Managed **(e).** Optimized
It is basically an audit.	It is basically an appraisal.
It is open to multi sector.	It is open to IT/ITES.
Follow set of standards to make success repeatable.	It emphasizes a process of continuous improvement.