By Shubh Thakare +919595939264

Unit 4.Metrics for Process and Projects

Metrics for Process and Projects :

What metrics are essential for a project's success?

How do you know which parameters are the most appropriate to define and measure the successful completion of a process?

What are some software development metrics examples?
Read on to find out all you need to know about Project & Process Metrics for Project Management.

Project and Process Metrics Classifying the process Metric Measurement

Key Project & Process Metric Groups

Project managers have a wide variety of metrics to choose from.

We can classify the most commonly used metrics into the following groups:

1. **Process Metrics**

   These are metrics that pertain to Process Quality. They are used to measure the efficiency and effectiveness of various processes.

2. **Project Metrics**

   These are metrics that relate to Project Quality. They are used to quantify defects, cost, schedule, productivity and estimation of various project resources and deliverables.

3. **Product Metrics**

   These are metrics that pertain to Product Quality. They are used to measure cost, quality, and the product's time-to-market.

4. **Organizational Metrics**

   These metrics measure the impact of organizational economics, employee satisfaction, communication, and organizational growth factors of the project.

5. **Software Development Metrics Examples**

   These metrics enable management to understand the quality of the software, the productivity of the development team, code complexity, customer satisfaction, agile process, and operational metrics.

1.Software Measurement

Software Measurement:
A measurement is an manifestation of the size, quantity, amount or dimension of a particular attributes 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. Software measurement process is defined and governed by ISO Standard.
Measures

Need of Software Measurement
Software is measured to:
1.Create the quality of the current product or process.
2.Anticipate future qualities of the product or process.
3.Enhance the quality of a product or process.
4.Regulate the state of the project in relation to budget and schedule.

Classification of Software Measurement:

There are 2 types of software measurement:

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

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

A software metric is a measure of software characteristics which are measurable or countable. Software metrics are valuable for many reasons, including measuring software performance, planning work items, measuring productivity, and many other uses. Within the software development process, many metrics are that are all connected.
Software metrics are similar to the four functions of management: Planning, Organization, Control, or Improvement.

Characteristics of software Metrics:

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

Classification of Software Metrics

Software metrics can be classified into three types as follows:

Product Metrics:

These are the measures of various characteristics of the software product. The two important software characteristics are:
• Size and complexity of software.
• Quality and reliability of software.
These metrics can be computed for different stages of SDLC.

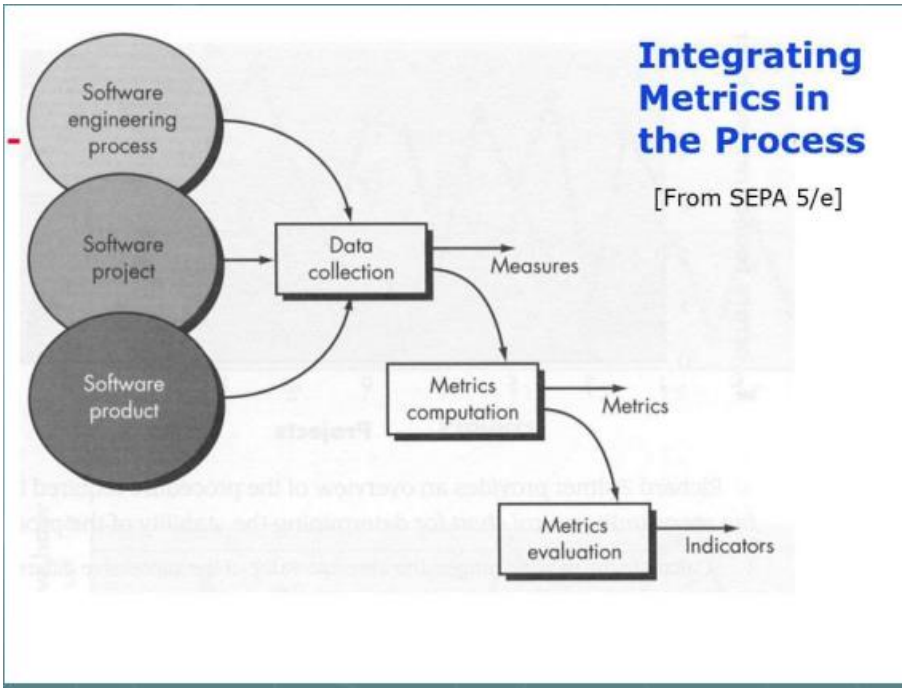
Process Metrics:

These are the measures of various characteristics of the software development process.
For example, the efficiency of fault detection. They are used to measure the characteristics of methods, techniques, and tools that are used for developing software.

Project Metrics:

Project matrix is describes the project characteristic and execution process.
•Number of software developer
•Staffing pattern over the life cycle of software
•Cost and schedule
•Productivity

By Shubh Thakare +919595939264

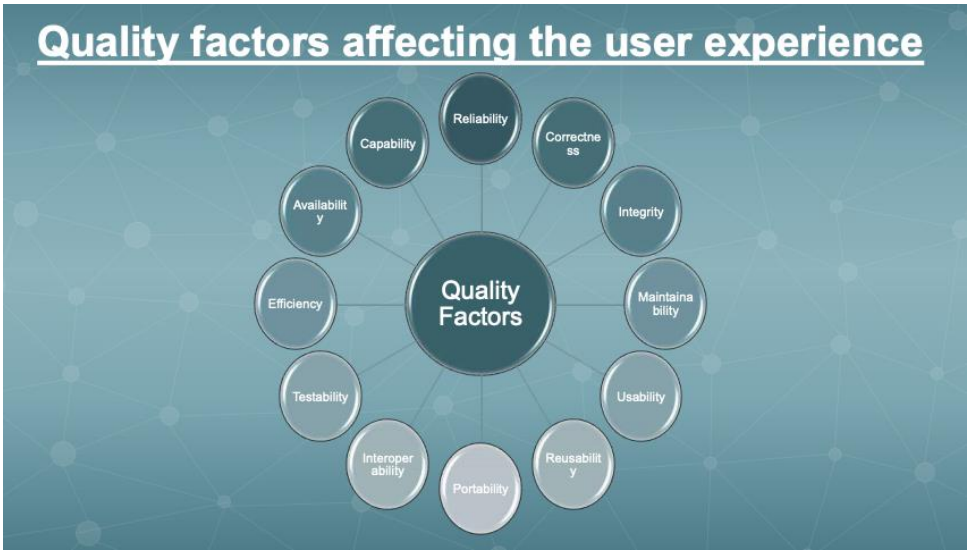


Metric for software Quality Software

Quality Metrics means measurement of attributes, pertaining to software quality along with its process of development. The term "software quality metrics" illustrate the picture of measuring the software qualities by recording the number of defects or security loopholes present in the software.

Why software Quality Metrics ?

• To define and categorised the elements in order to have better understanding of each and every process and attributes.
• To evaluate an assess each of these process and attribute against the given requirements and specification.
• Predicting and planning the next move with respect to software and business requirements .
•Improving the overall quality of the process and product , and subsequently of project.



By Shubh Thakare +919595939264

Metric for software Quality

2.Metrics for software quality.

Methodology Of Software Quality Metrics:

•Identify and prepare the list of possible requirements of quality, and subsequently, assigning direct metric, such as understanding, learning and operation time, to each of these requirements.
•Apply metrics framework, along with the cost-benefit analysis.
•Implementing metrics via collecting and defining data to compute metric values.
•Interpret and analyze the results, to ensure the fulfilment of requirements.
•Validate the metrics through validation methodology and thereafter proper documentation of the results.

Metric for software Quality

Features of good Software Quality Metrics
•Should be specific to measure the particular attribute or an attribute of greater importance.
•Comprehensive for wide variety of scenarios.
•Should not consider attributes that have already been measured by some other metric.
•Reliable to work similarly in all conditions.
•Should be easy and simple to understand and operate.

Software Quality Assurance

Quality of Design:
Quality of Design refers to the characteristics that designers specify for an item. The grade of materials, tolerances, and performance specifications that all contribute to the quality of design.

By Shubh Thakare +919595939264

Quality of conformance:
Quality of conformance is the degree to which the design specifications are followed during manufacturing. Greater the degree of conformance, the higher is the level of quality of conformance.

Software Quality:
Software Quality is defined as the conformance to explicitly state functional and performance requirements, explicitly documented development standards, and inherent characteristics that are expected of all professionally developed software

Quality Control:
Quality Control involves a series of inspections, reviews, and tests used throughout the software process to ensure each work product meets the requirements place upon it. Quality control includes a feedback loop to the process that created the work product.

Quality Assurance:
Quality Assurance is the preventive set of activities that provide greater confidence that the project will be completed successfully.

Risk management :

What is Risk?

"Tomorrow problems are today's risk." Hence, a clear definition of a "risk" is a problem that could cause some loss or threaten the progress of the project, but which has not happened yet.

These potential issues might harm cost, schedule or technical success of the project and the quality of our software device, or project team morale.

Risk Management is the system of identifying addressing and eliminating these problems before they can damage the project.

We need to differentiate risks, as potential issues, from the current problems of the project.

Different methods are required to address these two kinds of issues.

For example, staff storage, because we have not been able to select people with the right technical skills is a current problem, but the threat of our technical persons being hired away by the competition is a risk.

Risk Management

A software project can be concerned with a large variety of risks. In order to be adept to systematically identify the significant risks which might affect a software project, it is essential to classify risks into different classes. The project manager can then check which risks from each class are relevant to the project.

There are three main classifications of risks which can affect a software project:

1. Project risks
2. Technical risks
3. Business risks

1. Project risks: Project risks concern differ forms of budgetary, schedule, personnel, resource, and customer-related problems. A vital project risk is schedule slippage. Since the software is intangible, it is very tough to monitor and control a software project. It is very tough to control something which cannot be identified. For any manufacturing program, such as the manufacturing of cars, the plan executive can recognize the product taking shape.

2. Technical risks: Technical risks concern potential method, implementation, interfacing, testing, and maintenance issue. It also consists of an ambiguous specification, incomplete specification, changing specification, technical uncertainty, and technical obsolescence. Most technical risks appear due to the development team's insufficient knowledge about the project.

3. Business risks: This type of risks contain risks of building an excellent product that no one need, losing budgetary or personnel commitments, etc.

Other risk categories

1. **1. Known risks:** Those risks that can be uncovered after careful assessment of the project program, the business and technical environment in which the plan is being developed, and more reliable data sources (e.g., unrealistic delivery date)
2. **2. Predictable risks:** Those risks that are hypothesized from previous project experience (e.g., past turnover)
3. **3. Unpredictable risks:** Those risks that can and do occur, but are extremely tough to identify in advance.

Principle of Risk Management

1. **Global Perspective:** In this, we review the bigger system description, design, and implementation. We look at the chance and the impact the risk is going to have.
2. **Take a forward-looking view:** Consider the threat which may appear in the future and create future plans for directing the next events.
3. **Open Communication:** This is to allow the free flow of communications between the client and the team members so that they have certainty about the risks.
4. **Integrated management:** In this method risk management is made an integral part of project management.
5. **Continuous process:** In this phase, the risks are tracked continuously throughout the risk management paradigm.

A risk management plan's fundamental goal is to minimise or effectively eradicate the potential threats to a business, be it across an Operations, Marketing, or IT department.

Why Risk Management is Very Important

There are absolutely no cons to risk management if you have the correct strategy.

Benefits of a successful risk management strategy include:

- Prevention of Losses
- Decreased magnitude of losses.
- Resources are utilised more effectively to combat risk.
- Reduced number of reactive situations.
- Reassures stakeholders, partners, and clients that their data, investments, etc is safeguarded.
- Identify and promptly grasp new opportunities.
- Promote continuous improvement.

1.Reactive vs. Proactive Risk strategies

What is Proactive Strategy?

Proactive risk management is a concept that aims to reduce the odds of any accident or malicious attack occurring in the future by identifying each business' situation or processes to determine potential threats.

Enabling a proactive approach also involves correctly identifying drivers of risks to grasp the key issue along with assessing the likelihood and impact in order to prioritise risks, which makes creating a contingency plan a lot easier.

It is creative food for thought for those looking to protect their organisation's valuables and is currently the most common method of risk management strategy because of its effectiveness.

Why choose a Proactive Strategy?

The old adage of 'prevention is better than the cure' rings true with regards to a proactive strategy, and this can apply to an entire host of occupations who possess a risk management plan.

You'd argue that fire prevention is much more preferable than actual firefighting. This also applies to business, where retailers (etc) can safeguard their valuables from threats online by utilising a proactive risk management method, rather than manage the threat once it's reared its ugly head.

By Shubh Thakare +919595939264

What is Reactive Strategy?

A Reactive risk strategy is a response based approach, which is solely dependent on past incident evaluation and audit based findings. It begins once an 'incident' occurs, or once problems have been detected following on from an audit.

The incident is investigated, and measures are implemented to avoid similar events attacking the business in the future.

Why choose a Reactive Strategy?

It saves company time that is poured in to the research and analysis of risk mitigation, and forces a company's hand to adapt in certain situations, which can be risky.

Reactive concepts also save a lot of time and energy that a proactive approach would take. It is also argued that since we do not possess the gift of foresight, a situation that a proactive strategy has not prepared for may arise, in which case the strategy is rendered surplus to requirements.

Proactive vs. Reactive Which Strategy is Best?

Regardless of your business model, a proactive risk mitigation approach is customarily the defence method of choice when it comes to security.

A proactive risk strategy generally consists of focusing efforts on mitigating the risk of pre-emptive threat occurrences. This involves devising plans to protect critical assets via educating the business about the who, what, where, why, and how threats can potentially expose their valuable, sensitive assets.

By profiling each asset, businesses can also easily prioritise, plan and tackle incidents accordingly should they threaten to compromise their assets.

A Reactive risk management strategy possesses the inability to efficiently manage or eliminate threats

There is the small argument that too many resources and time can be put into planning for something that may never happen, but logic would dictate that by preemptively seeking out issues, you're effectively extinguishing the flames before the fire has started.

The application of a proactive risk approach is so beneficial in such universal circumstances, it's hard to see why people and businesses alike would neglect to employ any other strategy, especially one that would see themselves stuck in a continuous state of damage control

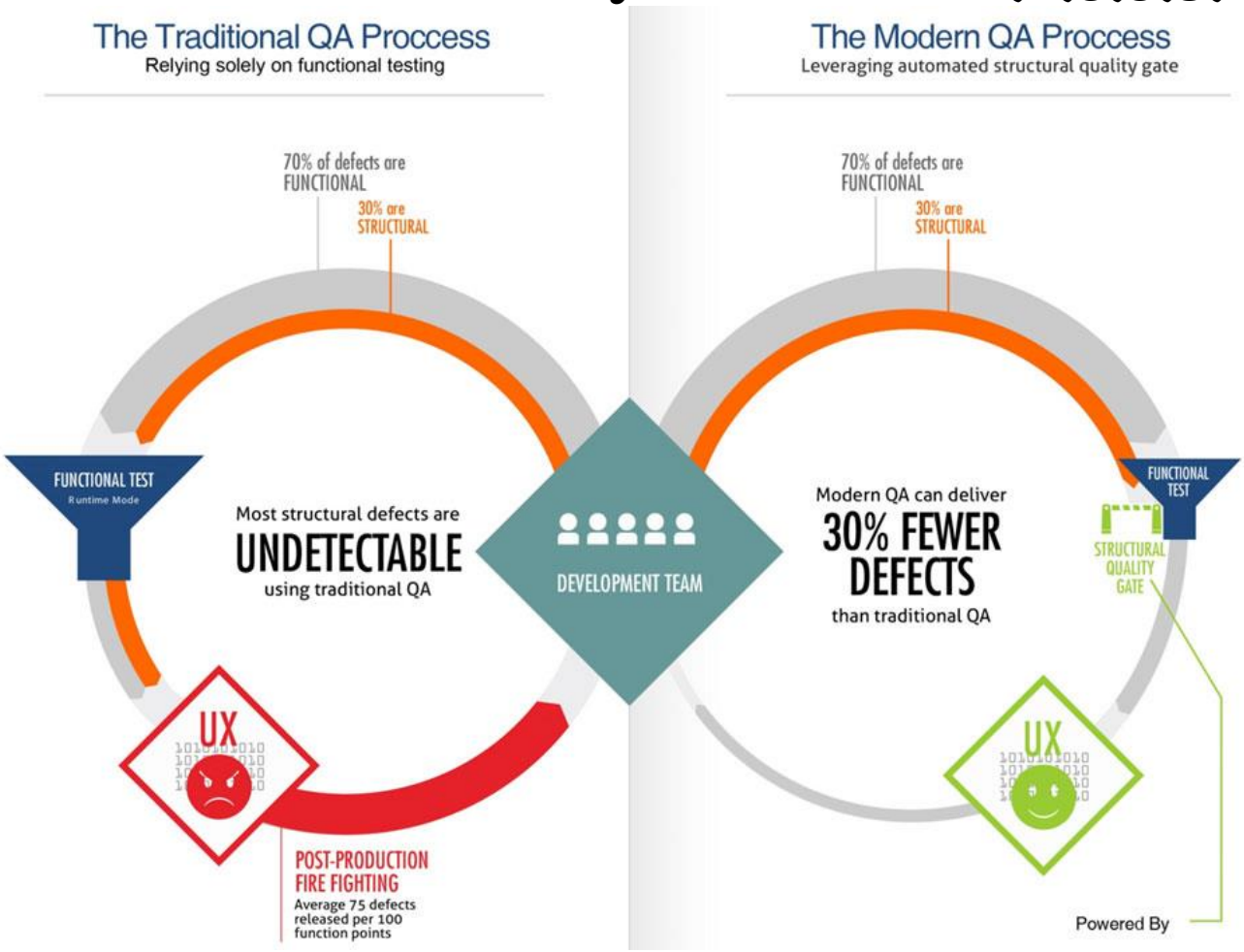
By Shubh Thakare +919595939264

2.Software Risks

Software risk encompasses the probability of occurrence for uncertain events and their potential for loss within an organization. Risk management has become an important component of software development as organizations continue to implement more applications across a multiple technology, multi-tiered environment. Typically, software risk is viewed as a combination of robustness, performance efficiency, security and transactional risk propagated throughout the system.

How to Reduce Software Risk

By Shubh Thakare +919595939264



Most organizations don't have a process to directly address the software risk that results from active custom software development. The traditional approach is to rely on testing – regression tests, system integration tests, performance tests and user integration tests. As you can see in the diagram, 30% of defects discovered in QA and live use are structural. And it is the structural defects that are the primary software risk exposure in the application lifecycle. Based on known software economics, that's 25 defects per function point that directly lead to software risk. Adding a structural quality gate to the QA process is imperative in order to measure and prevent software risk in mission critical systems. Most structural quality defects are actually **not** related to code quality issues, according to industry sources. It's a common misconception that code quality tools might address software risk. In reality, structural quality requires system level analysis in order to detect defects that pose software risk.

Prevention Is Key!

In a complex technology environment, it is not enough to deal with problems as they become apparent. Prevention is key to experiencing flawless performance and getting the most out of systems, applications, and your development team. Exposing the not so obvious weaknesses in an infrastructure by using dependable software risk analysis solutions ensures the proper identification of:

- System Vulnerabilities
- Compliance Issues
- Stability Problems
- Efficiency Weaknesses
- Performance Degradation
- Security Flaws

Are you struggling with pinpointing or managing potential problems in a complex IT environment?
Is your organization capable of finding system critical issues prior to executing an application?

3.Risk identification

Risk Identification

SOFTWARE ENGINEERING

Risk identification is a systematic attempt to specify threats to the project plan (estimates, schedule, resource loading, etc.). By identifying known and predictable risks, the project manager takes a first step toward avoiding them when possible and controlling them when necessary.

There are two distinct types of risks : generic risks and product-specific risks.
Generic risks are a potential threat to every software project.
Product-specific risks can be identified only by those with a clear understanding of the technology, the people, and the environment that is specific to the project at hand.
To identify product-specific risks, the project plan and the software statement of scope are examined and an answer to the following question is developed:
"What special characteristics of this product may threaten our project plan?"

One method for identifying risks is to create a risk item checklist. The checklist can be used for risk identification and focuses on some subset of known and predictable risks in the following generic subcategories:

• **Product size**—risks associated with the overall size of the software to be built or modified.

• **Business impact**—risks associated with constraints imposed by management or the marketplace.

• **Customer characteristics**—risks associated with the sophistication of the customer and the developer's ability to communicate with the customer in a timely manner.

• **Process definition**—risks associated with the degree to which the software process has been defined and is followed by the development organization.

• **Development environment**—risks associated with the availability and quality of the tools to be used to build the product.

• **Technology to be built**—risks associated with the complexity of the system to be built and the "newness" of the technology that is packaged by the system.

• **Staff size and experience**—risks associated with the overall technical and project experience of the software engineers who will do the work.

4.Risk projection

Risk Projection

SOFTWARE ENGINEERING

Risk projection, also called risk estimation, attempts to rate each risk in two ways—the likelihood or probability that the risk is real and the consequences of the problems associated with the risk, should it occur.

The project planner, along with other managers and technical staff, performs four risk projection activities:
(1) establish a scale that reflects the perceived likelihood of a risk,
(2) delineate the consequences of the risk,
(3) estimate the impact of the risk on the project and the product, and (4)note the overall accuracy of the risk projection so that there will be no misunderstandings.

Developing a Risk Table

A risk table provides a project manager with a simple technique for risk projection .A project team begins by listing all risks (no matter how remote) in the first column of the table. This can be accomplished with the help of the risk item checklists Each risk is categorized in the second column (e.g., PS implies a project size risk, BU implies a business risk). The probability of occurrence of each risk is entered in the

next column of the table. The probability value for each risk can be estimated by team members individually. Individual team members are polled in round-robin fashion until their assessment of risk probability begins to converge.

Next, the impact of each risk is assessed. Each risk component is assessed and an impact category is determined. The categories for each of the four risk components—performance, support, cost, and schedule—are averaged to determine an overall impact value.

Once the first four columns of the risk table have been completed, the table is sorted by probability and by impact. High-probability, high-impact risks percolate to the top of the table, and low-probability risks drop to the bottom. This accomplishes first-order risk prioritization.

The project manager studies the resultant sorted table and defines a cutoff line. The cutoff line (drawn horizontally at some point in the table) implies that only risks that lie above the line will be given further attention. Risks that fall below the line are re-evaluated to accomplish second-order prioritization. Risk impact and probability have a distinct influence on management concern. A risk factor that has a high impact but a very low probability of occurrence should not absorb a significant amount of management time. However, high-impact risks with moderate to high probability and low-impact risks with high probability should be carried forward into the risk analysis steps that follow.

5.Risk refinement

Risk Refinement

SOFTWARE ENGINEERING

During early stages of project planning, a risk may be stated quite generally. As time passes and more is learned about the project and the risk, it may be possible to refine the risk into a set of more detailed risks, each somewhat easier to mitigate, monitor, and manage.

One way to do this is to represent the risk in condition-transition-consequence (CTC) format . That is, the risk is stated in the following form: Given that <condition> then there is concern that (possibly) <consequence>.

Using the CTC format for the reuse risk noted in Section 6.4.2, we can write: Given that all reusable software components must conform to specific design standards and that some do not conform, then there is concern that (possibly) only 70 percent of the planned reusable modules may actually be integrated into the as-built system, resulting in the need to custom engineer the remaining 30 percent of components.

This general condition can be refined in the following manner:

Subcondition 1. Certain reusable components were developed by a third party with no knowledge of internal design standards.

Subcondition 2. The design standard for component interfaces has not been solidified and may not conform to certain existing reusable components.

Subcondition 3. Certain reusable components have been implemented in a language that is not supported on the target environment.

The consequences associated with these refined subconditions remains the same (i.e., 30 percent of software components must be customer engineered), but the refinement helps to isolate the underlying risks and might lead to easier analysis and response.

6.RMMM

R.M.M.M | Risk Mitigation(Avoidance /Reduce) Monitoring and Management:

R.M.M.M stands for risk mitigation, monitoring and management. There are three issues in strategy for handling the risk is

- Risk Avoidance
- Risk monitoring

- Risk management

Risk Mitigation:

- Risk mitigation means preventing the risk to occur (Risk Avoidance). Following are the steps to be taken for mitigating the risks.
- Communicate with the concerned staff to find of probable risk.
- Find out and eliminate all those causes that can create risk before the project starts.
- Develop a policy in an organization which will help to continue the project even though some staff leaves the organization.
- Everybody in the project team should be acquainted with the current development activity.
- Maintain the corresponding documents in timely manner. This documentation should be strictly as per the standards set by the organization.
- Conduct timely reviews in order to speed up the work.
- For conducting every critical activity during software development, provide the additional staff if required.

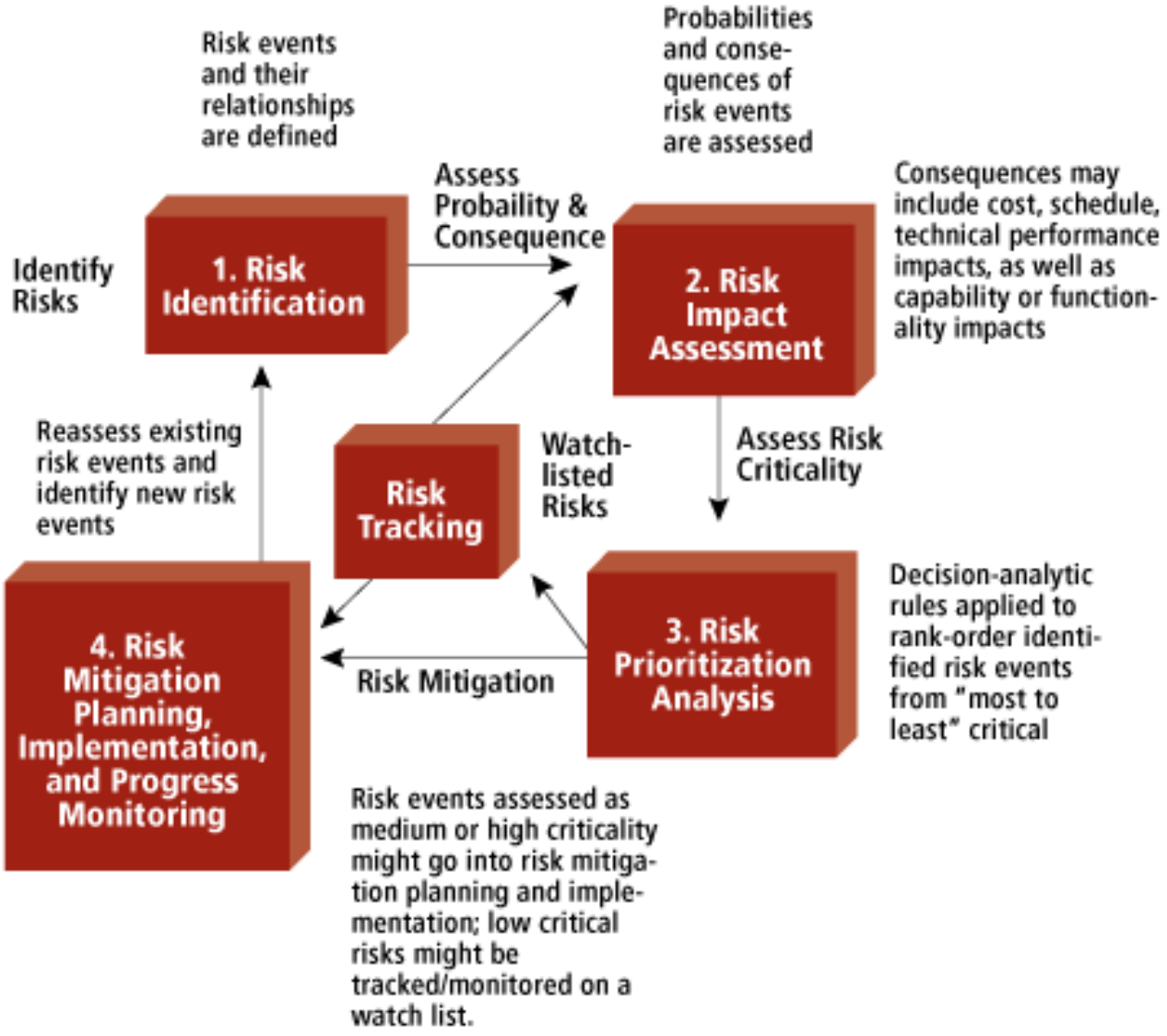
Risk Monitoring: (Manage By Project Manager)

- In risk monitoring process following things must be monitored by the project manager..
- The approach of the team members as pressure of project varies.
- The degree in which the team performs with the spirit of "team-work".
- The type of co-operation among the team members.The types of problems that are

occurring.Availability of jobs within and outside the organization.

- The project manager should monitor certain mitigation steps.
- For example: If the current development activity is monitored continuously then everybody in the team will get acquainted with current development activity.
- The objective of risk monitoring is to check whether the predicted risks really occur or not. To ensure the step defined to avoid the risk are applied properly or not. To gather the information which can be useful for analyzing the risk.

Risk management:



- Project manager perform this task when risk becomes a reality. If project manager is successful in applying the project mitigation effectively then it becomes very much easy to manage the risks.
- For example, Consider a scenario that many people are leaving the organization then if sufficient additional staff is available, if current development activity is know to everybody in the team, if latest and systematic documentation is available then any 'new comer' can easily understand current development activity. This will ultimately help in continuing the work without any interval.

By Shubh Thakare +919595939264

7.RMMM Plan.

R.M.M.M Plan:

- The R.M.M.M plan is a document in which all the risk analysis activities are described. Sometimes project manager includes this document as a part of overall project plan. Sometimes specific R.M.M.M plan is not created, however each risk can be described individually using risk information sheet. Typical template for R.M.M.M plan or risk information sheet can be,

Final Words: The Risk information sheet can be maintained by database systems. After documenting the risks using either R.M.M.M plan or risk information sheet the risk mitigation, monitoring and analysis activities are stopped.

The RMMM Plan

SOFTWARE ENGINEERING

A risk management strategy can be included in the software project plan or the risk management steps can be organized into a separate Risk Mitigation, Monitoring and Management Plan. The RMMM plan documents all work performed as part of risk analysis and is used by the **project manager** as part of the overall project plan.

Some software teams do not develop a formal RMMM document. Rather, each risk is documented individually using a risk information sheet . In most cases, the RIS is maintained using a database system, so that creation and information entry, priority ordering, searches, and other analysis may be accomplished easily.

Once RMMM has been documented and the project has begun, risk mitigation and monitoring steps commence. As we have already discussed, risk mitigation is a problem avoidance activity. Risk monitoring is a project tracking activity with three primary objectives:

(1) to assess whether predicted risks do, in fact, occur;

(2) to ensure that risk aversion steps defined for the risk are being properly applied; and

(3) to collect information that can be used for future risk analysis.

In many cases, the problems that occur during a project can be traced to more than one risk. Another job of risk monitoring is to attempt to allocate origin (what risk(s) caused which problemsthroughout the project).

Quality Management :

What is Software Quality Management?

Software Quality Management ensures that the required level of quality is achieved by submitting improvements to the product development process. SQA aims to develop a culture within the team and it is seen as everyone's responsibility.

Software Quality management should be independent of project management to ensure independence of cost and schedule adherences. It directly affects the process quality and indirectly affects the product quality.

Activities of Software Quality Management:

- **Quality Assurance -** QA aims at developing Organizational procedures and standards for quality at Organizational level.
- **Quality Planning -** Select applicable procedures and standards for a particular project and modify as required to develop a quality plan.
- **Quality Control -** Ensure that best practices and standards are followed by the software development team to produce quality products.

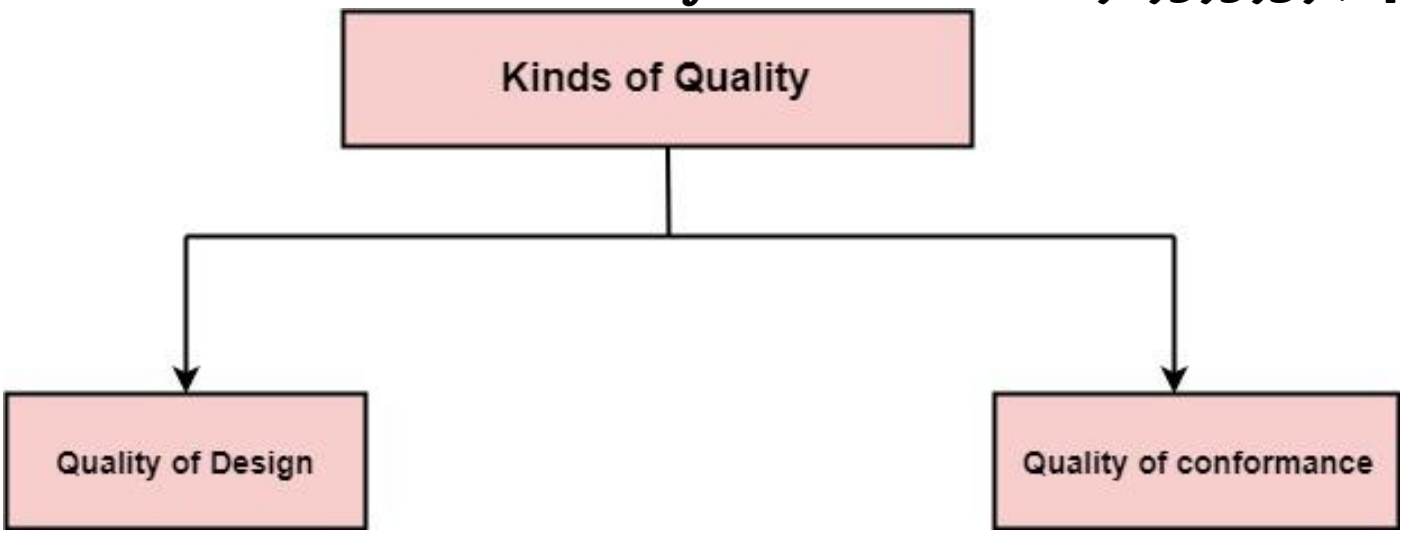
1.Quality concepts

What is Quality?

Quality defines to any measurable characteristics such as correctness, maintainability, portability, testability, usability, reliability, efficiency, integrity, reusability, and interoperability.

There are two kinds of Quality:

By Shubh Thakare +919595939264



Quality of Design: Quality of Design refers to the characteristics that designers specify for an item. The grade of materials, tolerances, and performance specifications that all contribute to the quality of design.

Quality of conformance: Quality of conformance is the degree to which the design specifications are followed during manufacturing. Greater the degree of conformance, the higher is the level of quality of conformance.

Software Quality: Software Quality is defined as the conformance to explicitly state functional and performance requirements, explicitly documented development standards, and inherent characteristics that are expected of all professionally developed software.

Quality Control: Quality Control involves a series of inspections, reviews, and tests used throughout the software process to ensure each work product meets the requirements place upon it. Quality control includes a feedback loop to the process that created the work product.

Quality Assurance: Quality Assurance is the preventive set of activities that provide greater confidence that the project will be completed successfully.

Quality Assurance focuses on how the engineering and management activity will be done?

As anyone is interested in the quality of the final product, it should be assured that we are building the right product.

Importance of Quality

We would expect the quality to be a concern of all producers of goods and services. However, the distinctive characteristics of software and in particular its intangibility and complexity, make special demands.

Increasing criticality of software: The final customer or user is naturally concerned about the general quality of software, especially its reliability. This is increasing in the case as organizations become more dependent on their computer systems and software is used more and more in safety-critical areas. For example, to control aircraft.

The intangibility of software: This makes it challenging to know that a particular task in a project has been completed satisfactorily. The results of these tasks can be made tangible by demanding that the developers produce 'deliverables' that can be examined for quality.

Accumulating errors during software development: As computer system development is made up of several steps where the output from one level is input to the next, the errors in the earlier ?deliverables? will be added to those in the later stages leading to accumulated determinable effects.

2.Software quality assurance

Software Quality Assurance (SQA) is simply a way to assure quality in the software. It is the set of activities which ensure processes, procedures as well as standards suitable for the project and implemented correctly.
Software Quality Assurance is a process which works parallel to development of a software. It focuses on improving the process of development of software so that problems can be prevented before they become a major issue. Software Quality Assurance is a kind of an Umbrella activity that is applied throughout the software process.

Software Quality Assurance have:

1. A quality management approach
2. Formal technical reviews
3. Multi testing strategy
4. Effective software engineering technology
5. Measurement and reporting mechanism

Major Software Quality Assurance Activities:

1. **SQA Management Plan:**
   Make a plan how you will carry out the sqa through out the project. Think which set of software engineering activities are the best for project.check level of sqa team skills.
2. **Set The Check Points:**
   SQA team should set checkpoints. Evaluate the performance of the project on the basis of collected data on different check points.
3. **Multi testing Strategy:**
   Do not depend on single testing approach. When you have lot of testing approaches available use them.
4. **Measure Change Impact:**
   The changes for making the correction of an error sometimes re introduces more errors keep the measure of impact of change on project. Reset the new change to change check the compatibility of this fix with whole project.
5. **Manage Good Relations:**
   In the working environment managing the good relation with other teams involved in the project development is mandatory. Bad relation of sqa team with programmers team will impact directly and badly on project. Don't play politics.

Benefits of Software Quality Assurance (SQA):

1. SQA produce high quality software.
2. High quality application saves time and cost.
3. SQA is beneficial for better reliability.
4. SQA is beneficial in the condition of no maintenance for long time.
5. High quality commercial software increase market share of company.

6. Improving the process of creating software.
7. Improves the quality of the software.

Disadvantage of SQA:

There are a number of disadvantages of quality assurance. Some of them include adding more resources, employing more workers to help maintain quality and so much more.

3. Software Reviews

Software Review

Software Review is systematic inspection of a software by one or more individuals who work together to find and resolve errors and defects in the software during the early stages of Software Development Life Cycle (SDLC). Software review is an essential part of Software Development Life Cycle (SDLC) that helps software engineers in validating the quality, functionality and other vital features and components of the software. It is a whole process that includes testing the software product and it makes sure that it meets the requirements stated by the client.

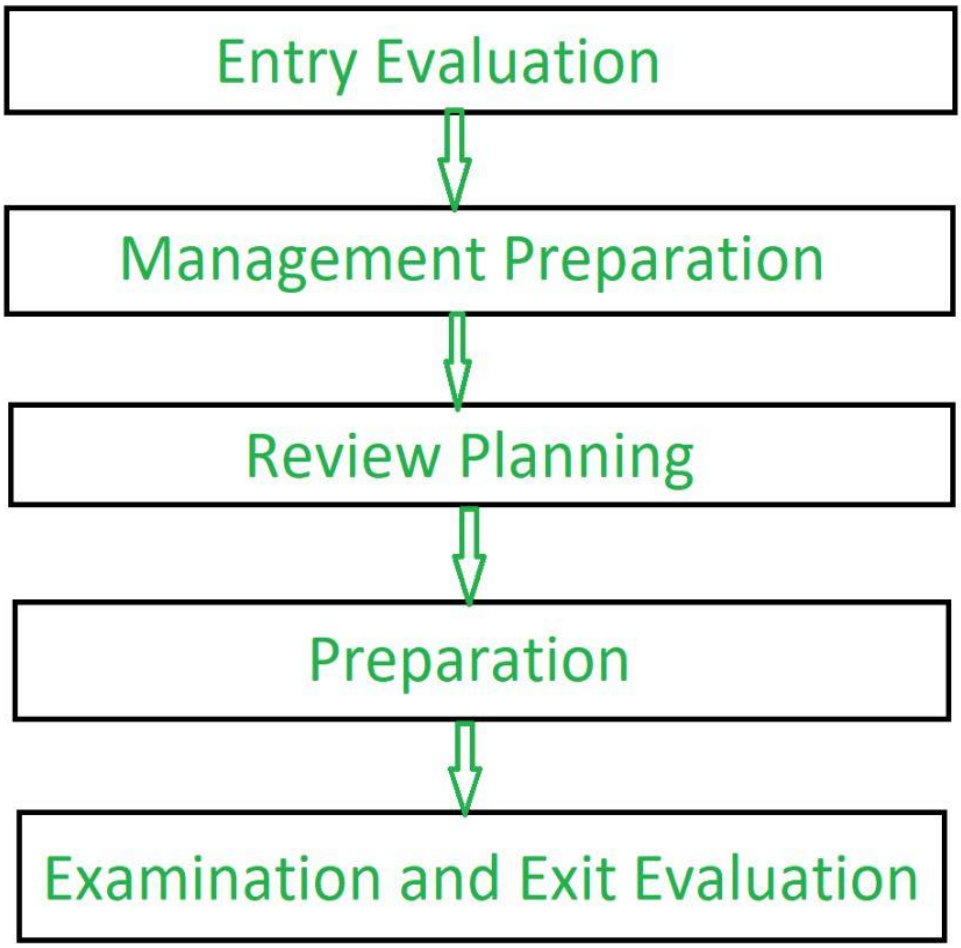
Usually performed manually, software review is used to verify various documents like requirements, system designs, codes, test plans and test cases.

Objectives of Software Review:

The objective of software review is:

1. To improve the productivity of the development team.
2. To make the testing process time and cost effective.
3. To make the final software with fewer defects.
4. To eliminate the inadequacies.

Process of Software Review:



Types of Software Reviews:
There are mainly 3 types of software reviews:

1. **Software Peer Review:**
   Peer review is the process of assessing the technical content and quality of the product and it is usually conducted by the author of the work product along with some other developers.
   Peer review is performed in order to examine or resolve the defects in the software, whose quality is also checked by other members of the team.
   Peer Review has following types:

   - **(i) Code Review:**
     Computer source code is examined in a systematic way.
   - **(ii) Pair Programming:**
     It is a code review where two developers develop code together at the same platform.
   - **(iii) Walkthrough:**
     Members of the development team is guided bu author and other interested parties and the participants ask questions and make comments about defects.
   - **(iv) Technical Review:**
     A team of highly qualified individuals examines the software product for

its client's use and identifies technical defects from specifications and standards.

- **(v) Inspection:**
  In inspection the reviewers follow a well-defined process to find defects.

2. **Software Management Review:**
   Software Management Review evaluates the work status. In this section decisions regarding downstream activities are taken.

3. **Software Audit Review:**
   Software Audit Review is a type of external review in which one or more critics, who are not a part of the development team, organize an independent inspection of the software product and its processes to assess their compliance with stated specifications and standards. This is done by managerial level people.

Advantages of Software Review:

- Defects can be identified earlier stage of development (especially in formal review).
- Earlier inspection also reduces the maintenance cost of software.
- It can be used to train technical authors.
- It can be used to remove process inadequacies that encourage defects.

4.Formal technical reviews

Formal Technical Review (FTR) in Software Engineering

Formal Technical Review (FTR) is a software quality control activity performed by software engineers.

Objectives of formal technical review (FTR):
Some of these are:

- Useful to uncover error in logic, function and implementation for any representation of the software.
- The purpose of FTR is to verify that the software meets specified requirements.
- To ensure that software is represented according to predefined standards.
- It helps to review the uniformity in software that is development in a uniform manner.
- To makes the project more manageable.

In addition, the purpose of FTR is to enable junior engineer to observer the analysis, design, coding and testing approach more closely. FTR also works to promote back up and continuity become familiar with parts of software they might not have seen otherwise.

Actually, FTR is a class of reviews that include walkthroughs, inspections, round robin reviews and other small group technical assessments of software. Each FTR is conducted as meeting and is considered successfully only if it is properly planned, controlled and attended.

The review meeting:

Each review meeting should be held considering the following constraints-

Involvement of people:

1. Between 3, 4 and 5 people should be involve in the review.
2. Advance preparation should occur but it should be very short that is at the most 2 hours of work for every person.
3. The short duration of the review meeting should be less than two hour. Gives these constraints, it should be clear that an FTR focuses on specific (and small) part of the overall software.

At the end of the review, all attendees of FTR must decide what to do.

1. Accept the product without any modification.
2. Reject the project due to serious error (Once corrected, another app need to be reviewed), or
3. Accept the product provisional (minor errors are encountered and are should be corrected, but no additional review will be required).

The decision was made, with all FTR attendees completing a sign-of indicating their participation in the review and their agreement with the findings of the review team.

Review reporting and record keeping :-

1. During the FTR, the reviewer actively records all issues that have been raised.
2. At the end of the meeting all these issues raised are consolidated and a review list is prepared.
3. Finally, a formal technical review summary report is prepared.

It answers three questions :-

1. What was reviewed ?
2. Who reviewed it ?
3. What were the findings and conclusions ?

Review guidelines :-

Guidelines for the conducting of formal technical reviews should be established in advance. These guidelines must be distributed to all reviewers, agreed upon, and then followed. A review that is unregistered can often be worse than a review that does not minimum set of guidelines for FTR.

1. Review the product, not the manufacture (producer).
2. Take written notes (record purpose)
3. Limit the number of participants and insists upon advance preparation.
4. Develop a checklist for each product that is likely to be reviewed.
5. Allocate resources and time schedule for FTRs in order to maintain time schedule.
6. Conduct meaningful training for all reviewers in order to make reviews effective.
7. Reviews earlier reviews which serve as the base for the current review being conducted.
8. Set an agenda and maintain it.

9. Separate the problem areas, but do not attempt to solve every problem notes.
10. Limit debate and rebuttal.

5.Statistical Software quality Assurance

Statistical Quality Assurance (SQA)

As brands and retailers experience growing demand for the latest consumer products, the resulting increase in production and batch sizes makes quality control more challenging for companies.

Traditional compliance testing techniques can sometimes provide limited pass/fail information, which results in insufficient measurements on the batch's quality control, identification of the root cause of failure results and overall quality assurance (QA) in the production process.

Intertek combines legal, customer and essential safety requirements to customize a workable QA process, called Statistical Quality Assurance (SQA). SQA is used to identify the potential variations in the manufacturing process and predict potential defects on a parts-per-million (PPM) basis. It provides a statistical description of the final product and addresses quality and safety issues that arise during manufacturing.

SQA consists of three major methodologies:

1. **Force Diagram** - A Force Diagram describes how a product should be tested. Intertek engineers base the creation of Force Diagrams on our knowledge of foreseeable use, critical manufacturing process and critical components that have high potential to fail.
2. **Test-to-Failure (TTF)** - Unlike any legal testing, TTF tells manufacturers on how many defects they are likely to find in every million units of output. This information is incorporated into the process and concludes if a product needs improvement in quality or if it is being over engineered, which will eventually lead to cost savings.
3. **Intervention** - Products are separated into groups according to the total production quantity and production lines. Each group then undergoes an intervention. The end result is measured by Z-value, which is the indicator of quality and consistency of a product to a specification. Intervention allows manufacturers to pinpoint a defect to a specific lot and production line; thus saving time and money in corrective actions.

6.Software reliability

Software Reliability (Error Free)

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.

7.The ISO 9000 quality standards.

ISO 9000 Certification

ISO (International Standards Organization) is a group or consortium of 63 countries established to plan and fosters standardization. ISO declared its 9000 series of standards in 1987. It serves as a reference for the contract between independent parties. The ISO 9000 standard determines the guidelines for maintaining a quality system. The ISO standard mainly addresses operational methods and organizational methods such as responsibilities, reporting, etc. ISO 9000 defines a set of guidelines for the production process and is not directly concerned about the product itself.

Types of ISO 9000 Quality Standards

By Shubh Thakare +919595939264

The ISO 9000 series of standards is based on the assumption that if a proper stage is followed for production, then good quality products are bound to follow automatically. The types of industries to which the various ISO standards apply are as follows.

1. **ISO 9001:** This standard applies to the organizations engaged in design, development, production, and servicing of goods. This is the standard that applies to most software development organizations.
2. **ISO 9002:** This standard applies to those organizations which do not design products but are only involved in the production. Examples of these category industries contain steel and car manufacturing industries that buy the product and plants designs from external sources and are engaged in only manufacturing those products. Therefore, ISO 9002 does not apply to software development organizations.
3. **ISO 9003:** This standard applies to organizations that are involved only in the installation and testing of the products. For example, Gas companies.

How to get ISO 9000 Certification?

An organization determines to obtain ISO 9000 certification applies to ISO registrar office for registration. The process consists of the following stages:

By Shubh Thakare +919595939264

ISO 9000 Certification

1. **Application:** Once an organization decided to go for ISO certification, it applies to the registrar for registration.
2. **Pre-Assessment:** During this stage, the registrar makes a rough assessment of the organization.
3. **Document review and Adequacy of Audit:** During this stage, the registrar reviews the document submitted by the organization and suggest an improvement.
4. **Compliance Audit:** During this stage, the registrar checks whether the organization has compiled the suggestion made by it during the review or not.
5. **Registration:** The Registrar awards the ISO certification after the successful completion of all the phases.
6. **Continued Inspection:** The registrar continued to monitor the organization time by time.