# UNIT-5(Metrics for Process and Products)

**Measurement in Software Engineering**

Software Engineering involves managing, costing, planning, modeling, analyzing, specifying, designing, implementing, testing, and maintaining software products.

For most of the development projects,

* We fail to set measurable targets for our software products
* We fail to understand and quantify the component cost of software projects
* We do not quantify or predict the quality of the products we produce

Thus, for controlling software products, measuring the attributes is necessary. Every measurement action must be motivated by a particular goal or need that is clearly defined and easily understandable. The measurement objectives must be specific, tried to what managers, developers and users need to know. Measurement is required to assess the status of the project, product, processes, and resources.

In software engineering, measurement is essential for the following three basic activities −

* To understand what is happening during development and maintenance
* To control what is happening in the project
* To improve processes and goals

**Software quality metrics** are standards used to measure the attributes of a software product that contribute to its quality. These metrics can be categorized into different types, each focusing on a specific aspect of quality. Here are some of the key metrics for software quality:

1. [**Code Quality**: This includes both quantitative metrics like the number of lines, complexity, functions, rate of bugs generation, and qualitative metrics like readability, code clarity, efficiency, and maintainability1](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/).
2. **Reliability**: Reliability metrics measure the stability of software and its ability to perform under expected conditions. [Common reliability metrics include Mean Time Between Failure (MTBF) and Mean Time To Repair (MTTR)](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/)[1](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/).
3. **Performance**: Performance metrics assess how well the software executes its functions in terms of speed and resource utilization. [They determine whether the software fulfills user requirements efficiently](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/)[1](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/).
4. **Usability**: Usability metrics evaluate how user-friendly the software is. [They measure the end-user’s satisfaction and ease of use](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/)[1](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/).
5. [**Correctness**: Correctness metrics check if the software functions as intended without errors and satisfies user needs](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/)[1](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/).
6. **Maintainability**: These metrics look at how easily software can be maintained and upgraded. [They include factors like the time required to adapt to new features and performance in changing environments](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/)[1](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/).
7. [**Integrity**: Integrity metrics assess the ease of integration with other software and the control over unauthorized access, which is crucial for cybersecurity](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/)[1](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/).
8. [**Security**: Security metrics focus on the software’s ability to protect against unauthorized access and data breaches](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/)[1](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/).
9. [**Defect Density**: This measures the number of defects relative to the software size, such as lines of code or function points](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/)[2](https://www.tutorialspoint.com/software_quality_management/software_quality_management_metrics.htm).
10. [**Customer Satisfaction**: This is often measured through surveys and can include metrics like the percentage of satisfied or dissatisfied customers](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/)[2](https://www.tutorialspoint.com/software_quality_management/software_quality_management_metrics.htm).

These metrics help stakeholders understand and improve the quality of software products throughout their development and maintenance lifecycle. [For a more detailed explanation of each metric and how they are applied in software engineering, you can refer to resources](https://www.geeksforgeeks.org/measuring-software-quality-using-quality-metrics/)

# What is Risk?

Aa "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.

## What Is Risk Management?

**Risk Management** is the system of identifying addressing and eliminating these problems before they can damage the project.When developing a risk management program, a company typically goes through the following step.

### Risk Identification

The first step is to identify the risks that might threaten your business model or business continuity. The below list is not exhaustive, but it does give you a sense of the threats you should be looking for:

* Operational risk
* Financial risk
* Reputational risk
* Compliance, legal, or regulatory risk

### Risk Assessment

Once you identify the risks that might strike your organization, you need to assess the actual likelihood of the risk happening and the possible harm it might cause. This lets you organize your most pressing risks by priority, so you can allocate risk management resources more efficiently.

### Risk Treatment (acceptance, avoidance, mitigation, transference)

You can treat different risks in different ways. For example, some companies might decide to accept certain cyber security risks, while others might transfer the potential damage (say, by taking out a cyber insurance policy). Still others might avoid the risk entirely (perhaps by not using a certain type of IT asset), and others might mitigate the risk by taking certain precautions to make the risk less harmful.

### Risk Monitoring

Risks evolve over time. So once you decide on a risk treatment, you must still monitor your risks to see whether they’ve become better, worse, or gone away entirely. Companies might use a blend of internal controls and external providers to enable active threat monitoring from all risk factors, especially when launching new products or new markets.

### Reactive risk management could mean the following:

* Preventing potential risks from becoming incidents
* Mitigating damage from incidents
* Stopping small threats from worsening
* Continuing critical business functions despite incidents
* Evaluating each incident to solve its root cause
* Monitoring to assure that the incident does not recur
* proactive risk management strategies include:
* Identifying existing risks to the enterprise, business unit, or project
* Developing a risk response
* Prioritizing identified risks according to the magnitude of their threat
* Analyzing risks to determine the best treatment for each
* Implementing controls necessary to prevent hazards from becoming threats or incidents
* Monitoring the threat environment continuously

## Reactive Risk Management

One fundamental point about reactive risk management is that the disaster or threat must occur before management responds. In contrast, proactive risk management is about taking preventative measures before the event to decrease its severity. That’s a good thing to do.

### Proactive Risk Management

As the name suggests, proactive risk management means that you identify risks before they happen and figure out ways to avoid or alleviate the risk. It seeks to reduce the hazard’s risk potential or (even better) prevent the threat altogether.

A good example is vulnerability testing and remediation. Any organization of appreciable size is likely to have vulnerabilities in its software that attackers could find and exploit. So regular testing can find and patch those vulnerabilities to eliminate that threat.

A proactive management strategy gives you more control over your risk management. For example, you can decide which issues should be top priorities and what potential damage you will accept.

Proactive management also involves constantly monitoring your systems, risk processes, cybersecurity, competition, business trends, and so forth. Understanding the level of risk before an event allows you to instruct your employees on how to mitigate them.

A proactive approach, however, implies that each risk is constantly monitored. It also requires regular risk reviews to update your current risk profile and to identify new risks affecting the company. This approach drives management to be constantly aware of the direction of those risks.

RMMM

A risk management strategy can be defined as a software project plan or the risk management steps. It 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.

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 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.

**Effective strategy must consider three issues:**

* risk avoidance
* risk monitoring
* risk management and contingency planning.

**THE RMMM PLAN**

Risk Mitigation, Monitoring and Management Plan (RMMM) – documents all work performed as part of risk analysis and is used by the project manager as part of the overall project plan.RIS is maintained using a database system, so that creation and information entry, priority ordering, searches, and other analysis may be accomplished easily. Risk monitoring is a project tracking activity

**Three primary objectives:**

* assess whether predicted risks do, in fact, occur
* ensure that risk aversion steps defined for the risk are being properly applied
* collect information that can be used for future risk analysis.

## Software Quality

Software Quality shows how good and reliable a product is. To convey an associate degree example, think about functionally correct software. It performs all functions as laid out in the [SRS document](https://www.geeksforgeeks.org/software-requirement-specification-srs-document-checklist/).

## Factors of Software Quality

The modern read of high-quality associates with software many quality factors like the following:

**1.Portability:** A software is claimed to be transportable, if it may be simply created to figure in several package environments, in several machines, with alternative code merchandise, etc.

1. **Usability:** A software has smart usability if completely different classes of users (i.e. knowledgeable and novice users) will simply invoke the functions of the merchandise.
2. **Reusability:** A software has smart reusability if completely different modules of the merchandise will simply be reused to develop new merchandise.
3. **Correctness:** Software is correct if completely different needs as laid out in the SRS document are properly enforced.
4. **Maintainability:** A software is reparable, if errors may be simply corrected as and once they show up, new functions may be simply added to the merchandise, and therefore the functionalities of the merchandise may be simply changed, etc
5. **Reliability:**Software is more reliable if it has fewer failures. Since software engineers do not deliberately plan for their software to fail, reliability depends on the number and type of mistakes they make. Designers can improve reliability by ensuring the software is easy to implement and change, by testing it thoroughly, and also by ensuring that if failures occur, the system can handle them or can recover easily.
6. **Efficiency.** The more efficient software is, the less it uses of CPU-time, memory, disk space,[network bandwidth](https://www.geeksforgeeks.org/introduction-to-bandwidth/), and other resources. This is important to customers in order to reduce their costs of running the software, although with today’s powerful computers, CPU time, memory and disk usage are less of a concern than in years gone by.

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.

# 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 are suitable for the project and implemented correctly.

Software Quality Assurance is a process which works parallel to [Software Development.](https://www.geeksforgeeks.org/what-is-software-development/) 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 Umbrella activity that is applied throughout the[software process.](https://www.geeksforgeeks.org/software-processes-in-software-engineering/)

## ****Software Quality Assurance (SQA) encompasse****s

SQA process Specific quality assurance and quality control tasks (including technical reviews and a multitiered testing strategy) Effective software engineering practice (methods and tools) Control of all software work products and the changes made to them a procedure to ensure compliance with[software development](https://www.geeksforgeeks.org/software-development/?ref=lbp) standards (when applicable) measurement and reporting mechanisms

## ****Elements of Software Quality Assurance (SQA)****

1. **Standards:**The IEEE, ISO, and other standards organizations have produced a broad array of software engineering standards and related documents. The job of SQA is to ensure that standards that have been adopted are followed, and all work products conform to them.
2. **Reviews and audits:** Technical reviews are a quality control activity performed by software engineers for software engineers. Their intent is to uncover errors. Audits are a type of review performed by SQA personnel (people employed in an organization) with the intent of ensuring that quality guidelines are being followed for software engineering work.
3. **Testing:** [Software testing](https://www.geeksforgeeks.org/software-testing-basics/) is a quality control function that has one primary goal—to find errors. The job of SQA is to ensure that testing is properly planned and efficiently conducted for primary goal of software.
4. **Error/defect collection and analysis**: SQA collects and analyzes error and defect data to better understand how errors are introduced and what software engineering activities are best suited to eliminating them.
5. **Change management:** SQA ensures that adequate change management practices have been instituted.
6. **Education:** Every software organization wants to improve its software engineering practices. A key contributor to improvement is education of software engineers, their managers, and other stakeholders. The SQA organization takes the lead in software process improvement which is key proponent and sponsor of educational programs.
7. **Security management:**SQA ensures that appropriate process and technology are used to achieve software security.
8. **Safety:** SQA may be responsible for assessing the impact of software failure and for initiating those steps required to reduce risk.
9. [**Risk management**](https://www.geeksforgeeks.org/software-risk-analysis/): The SQA organization ensures that risk management activities are properly conducted and that risk-related contingency plans have been established.

## ****Software Quality Assurance (SQA) focuses on****

* software’s portability
* software’s usability
* software’s reusability
* software’s correctness
* software’s maintainability
* software’s error control

## ****Software Quality Assurance (SQA) has****

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 (SQA) Activities****

1. **SQA Management Plan:** Make a plan for how you will carry out the SQA throughout the project. Think about which set of [software engineering](https://www.geeksforgeeks.org/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. **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 check the compatibility of this fix with whole project.
4. **Multi testing Strategy:** Do not depend on a single testing approach. When you have a lot of testing approaches available use them.
5. **Manage Good Relations:** In the working environment managing good relations 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.
6. **Maintaining records and reports :** Comprehensively document and share all QA records, including test cases, defects, changes, and cycles, for stakeholder awareness and future reference.
7. **Reviews software engineering activities :** The SQA group identifies and documents the processes. The group also verifies the correctness of software product.
8. **Formalize deviation handling :** Track and document software deviations meticulously. Follow established procedures for handling variances.

## ****Benefits of Software Quality Assurance (SQA)****

1. SQA produces 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 a 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.
8. It cuts maintenance costs. Get the release right the first time, and your company can forget about it and move on to the next big thing. Release a product with chronic issues, and your business bogs down in a costly, time-consuming, never-ending cycle of repairs.

## ****Disadvantage of Software Quality Assurance (SQA)****

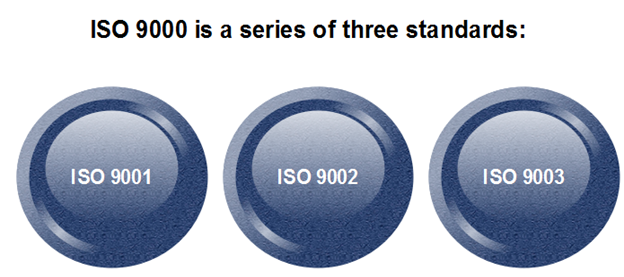
There are a number of disadvantages of quality assurance.

* **Cost:**Some of them include adding more resources, which cause the more budget its not, Addition of more resources For betterment of the product.
* **Time Consuming:**Testing and Deployment of the project taking more time which cause delay in the project.
* **Overhead:**Software Quality Assurance (SQA) will create depends on the administrator it will make slow the speed of the project.

# 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



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:



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.