Quality measurement

There is a number of metrics available based on which software quality is measured. But among them, there are a few most useful metrics which are essential in software quality measurement. They are –

1. Code Quality
2. Reliability
3. Performance
4. Usability
5. Correctness
6. Maintainability
7. Integrity
8. Security

1. Code Quality – Code quality metrics measure the quality of code used for software project development. Maintaining the software code quality by writing Bug-free and semantically correct code is very important for good software project development. In code quality, both Quantitative metrics like the number of lines, complexity, functions, rate of bugs generation, etc, and Qualitative metrics like readability, code clarity, efficiency, and maintainability, etc are measured.
2. Reliability – Reliability metrics express the reliability of software in different conditions. The software is able to provide exact service at the right time or not checked. Reliability can be checked using Mean Time Between Failure (MTBF) and Mean Time To Repair (MTTR).
3. Performance – Performance metrics are used to measure the performance of the software. Each software has been developed for some specific purposes. Performance metrics measure the performance of the software by determining whether the software is fulfilling the user requirements or not, by analyzing how much time and resource it is utilizing for providing the service.
4. Usability – Usability metrics check whether the program is user-friendly or not. Each software is used by the end-user. So it is important to measure that the end-user is happy or not by using this software.
5. Correctness – Correctness is one of the important software quality metrics as this checks whether the system or software is working correctly without any error by satisfying the user. Correctness gives the degree of service each function provides as per developed.
6. Maintainability – Each software product requires maintenance and up-gradation. Maintenance is an expensive and time-consuming process. So if the software product provides easy maintainability then we can say software quality is up to mark. Maintainability metrics include the time required to adapt to new features/functionality, Mean Time to Change (MTTC), performance in changing environments, etc.
7. Integrity – Software integrity is important in terms of how much it is easy to integrate with other required software which increases software functionality and what is the control on integration from unauthorized software's which increases the chances of cyberattacks.
8. Security – Security metrics measure how secure the software is. In the age of cyber terrorism, security is the most essential part of every software. Security assures that there are

no unauthorized changes, no fear of cyber attacks, etc when the software product is in use by the end-user.

Metrics, Measurement & Analysis:
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:

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.
- Enable data-driven decision-making in project planning and control.
- Identify bottlenecks and areas for improvement to drive process improvement activities.
- Ensure that industry standards and regulations are followed.
- Give software products and processes a quantitative basis for evaluation.
- Enable the ongoing improvement of software development practices.

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 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:

1. **Quantitative:** Metrics must possess quantitative nature. It means metrics can be expressed in values.
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.
4. **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. Examples include lines of code, cyclomatic complexity, code coverage, defect density, and code maintainability index.
2. **Process Metrics:** Process metrics pay particular attention to enhancing the long-term process of the team or organization. Examples include effort variance, schedule variance, defect injection rate, and lead time.

3. **Project Metrics:** The project matrix describes the project characteristic and execution process. Examples include effort estimation accuracy, schedule deviation, cost variance, and productivity.
    - 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.