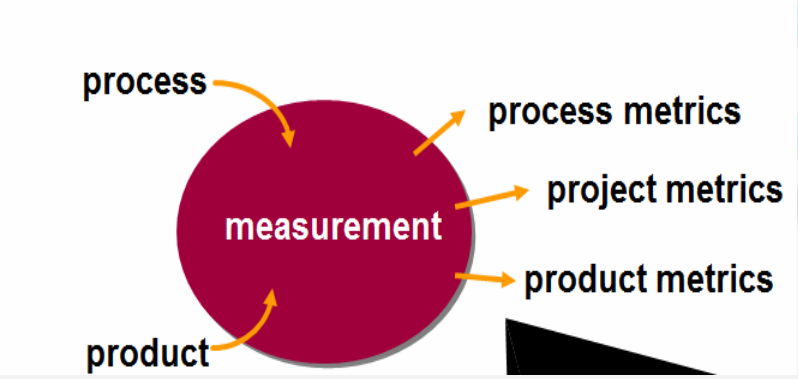
QUESTIONS AND ANSWERS ON SOFTWARE PROCESS AND PRODUCT METRICS

1) What are metrics?
Ans: Software Process and Product Metrics are quantitative measures.

- They are a management tool.
- They offer insight into the effectiveness of the software process and the projects that are conducted using the process as a framework.
- Basic quality and productivity data are collected.
- These data are analysed, compared against past averages, and assessed.
- The goal is to determine whether quality and productivity improvements have occurred.
- The data can also be used to pinpoint problem areas.
- Remedies can then be developed and the software process can be improved.

2) Need for Software Metrics:



To **characterize** in order to

- Gain an understanding of processes, products, resources, and environments.
- Establish baselines for comparisons with future assessments

To **evaluate** in order to

- Determine status with respect to plans

To **predict** in order to

- Gain understanding of relationships among processes and products.
- Build models of these relationships

To **improve** in order to

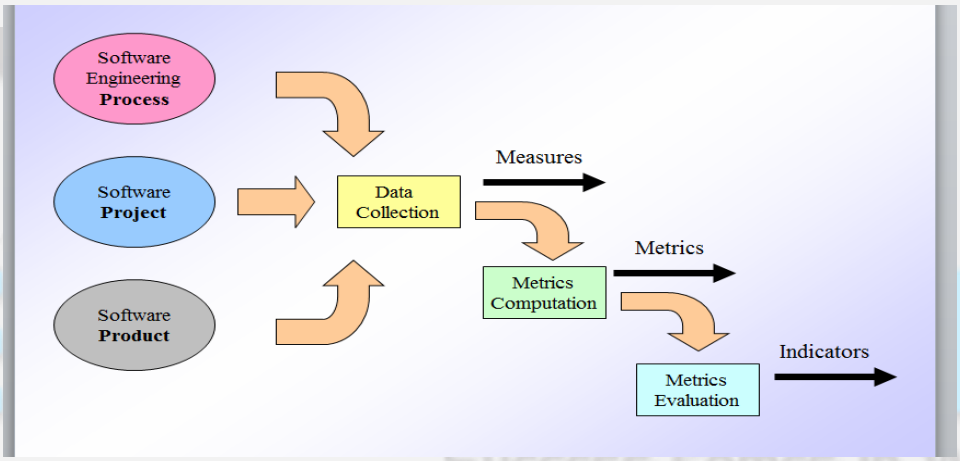
- Identify roadblocks, root causes, inefficiencies, and other opportunities for improving product quality and process performance.

3) What are Process Metrics?
Ans: Process metrics are collected across all projects and over long periods of time.

- They are used for making strategic decisions.
- The intent is to provide a set of process indicators that lead to long-term software process improvement.

The only way to know how/where to improve any process is to

1. Measure specific attributes of the process.
2. Develop a set of meaningful metrics based on these attributes.
3. Use the metrics to provide indicators that will lead to a strategy for improvement.



4) How can we measure the effectiveness of a Process?
Ans: We measure the effectiveness of a process by deriving a set of metrics based on outcomes of the process such as:

- Errors uncovered before release of the software.
- Defects delivered to and reported by the end users.
- Work products delivered.
- Human effort expended.
- Calendar time expended.
- Conformance to the schedule.
- Time and effort to complete each generic activity.

5) What is Product Metrics?
Ans: They focus on the quality of deliverables. Product metrics are combined across several projects to produce process metrics.
Metrics for the product:

- Measures of the Analysis Model.
- Complexity of the Design Model
- Code metrics.

Furthermore, Complexity of the Design Model is classified as-

1. Internal algorithmic complexity.
2. Architectural complexity.
3. Data flow complexity.

6) What are the attributes of a software metrics?
Ans: Following are the attributes of a software metrics-

1. Simple and computable. It should be relatively easy to learn how to derive the metric, and its computation should not demand inordinate effort or time.
2. Empirically and intuitively persuasive. The metric should satisfy the engineer's intuitive notions about the product attribute under consideration
3. Consistent and objective. The metric should always yield results that are unambiguous.
4. Consistent in its use of units and dimensions. The mathematical computation of the metric should use measures that do not lead to bizarre combinations of unit.
5. Programming language independent. Metrics should be based on the analysis model, the design model, or the structure of the program itself.
6. An effective mechanism for quality feedback. That is, the metric should provide a software engineer with information that can lead to a higher quality end product.

7) Explain Normalization for Metrics.
Ans: It tells us how an organization combines metrics that come from different individuals or projects.

- Depend on the size and complexity of the project.
- Normalization: compensate for complexity aspects particular to a product

8) Explain Normalization approaches.
Ans: Normalization approaches:

1. Size oriented (lines of code approach): Derived by normalizing quality and/or productivity measures by considering the size of the software produced.

- Thousand lines of code (KLOC) are often chosen as the normalization value.
- Metrics include

1. Errors per KLOC - Errors per person-month.
2. Defects per KLOC - KLOC per person-month.
3. Rs per KLOC - Rs per page of documentation.
4. Pages of documentation per KLOC.

- Size-oriented metrics are not universally accepted as the best way to measure the software process.
- Opponents argue that KLOC measurements-

1. Are dependent on the programming language.
2. Penalize well-designed but short programs.
3. Cannot easily accommodate nonprocedural languages.
4. Require a level of detail that may be difficult to achieve

2. Function oriented (function point approach): Function-oriented metrics use a measure of the functionality delivered by the application as a normalization value

- Most widely used metric of this type is the function point: FP = count total * [0.65 + 0.01 * sum (value adj. factors)]

- Function point values on past projects can be used to compute, for example, the average number of lines of code per function point (e.g., 60).

9) List importance of Software Metrics.
Ans: Importance of software Metrics:

- Most software developers do not measure, and most have little desire to begin.
- Establishing a successful company-wide software metrics program can be a multi-year effort. But if we do not measure, there is no real way of determining whether we are improving.
- Measurement is used to establish a process baseline from which improvements can be assessed.
- Software metrics help people to develop better project estimates, produce higher-quality systems, and get products out the door on time.

10) How can we compute FP?

Ans: FP can be computed as by following steps-

1. Analyze information domain of the application and develop counts.
2. Establish count for input domain and system.
3. Weight each count by assessing complexity.
4. Assign level of complexity or weight to each count.
5. Assess the influence of global factors that affect the application.
6. Grade significance of external factors, such reuse, OS, concurrency.
7. Compute Function Point.

FP = count total * [0.65 + 0.01 * sum (value adj. factors)]