**1. Lines of Code (LOC):** As the name suggest, LOC count the total number of lines of source code in a project. The units of LOC are:

* KLOC- Thousand lines of code
* NLOC- Non comment lines of code
* KDSI- Thousands of delivered source instruction

The size is estimated by comparing it with the existing systems of same kind. The experts use it to predict the required size of various components of software and then add them to get the total size.

**Advantages:**

* Universally accepted and is used in many models like COCOMO.
* Estimation is closer to developer’s perspective.
* Simple to use.

**Disadvantages:**

* Different programming languages contains different number of lines.
* No proper industry standard exist for this technique.
* It is difficult to estimate the size using this technique in early stages of project.

**2. Number of entities in ER diagram:**[ER model](https://www.geeksforgeeks.org/database-management-system-er-model/) provides a static view of the project. It describes the entities and its relationships. The number of entities in ER model can be used to measure the estimation of size of project. Number of entities depends on the size of the project. This is because more entities needed more classes/structures thus leading to more coding.

**Advantages:**

* Size estimation can be done during initial stages of planning.
* Number of entities is independent of programming technologies used.

**Disadvantages:**

* No fixed standards exist. Some entities contribute more project size than others.
* Just like FPA, it is less used in cost estimation model. Hence, it must be converted to LOC.

**3. Total number of processes in detailed data flow diagram:** Data Flow Diagram(DFD) represents the functional view of a software. The model depicts the main processes/functions involved in software and flow of data between them. Utilization of number of functions in DFD to predict software size. Already existing processes of similar type are studied and used to estimate the size of the process. Sum of the estimated size of each process gives the final estimated size.

**Advantages:**

* It is independent of programming language.
* Each major processes can be decomposed into smaller processes. This will increase the accuracy of estimation

**Disadvantages:**

* Studying similar kind of processes to estimate size takes additional time and effort.
* All software projects are not required to construction of DFD.

**4. Function Point Analysis:** In this method, the number and type of functions supported by the software are utilized to find FPC(function point count). The steps in function point analysis are:

* Count the number of functions of each proposed type.
* Compute the Unadjusted Function Points(UFP).
* Find Total Degree of Influence(TDI).
* Compute Value Adjustment Factor(VAF).
* Find the Function Point Count(FPC).

**Software Quality Assurance**

**Software Quality Assurance** (SQA) is a set of activities for ensuring quality in software engineering processes. It ensures that developed software meets and complies with the defined or standardized quality specifications. SQA is an ongoing process within the Software Development Life Cycle (SDLC) that routinely checks the developed software to ensure it meets the desired quality measures.

SQA practices are implemented in most types of software development, regardless of the underlying software development model being used. SQA incorporates and implements software testing methodologies to test the software. Rather than checking for quality after completion, SQA processes test for quality in each phase of development, until the software is complete. With SQA, the software development process moves into the next phase only once the current/previous phase complies with the required quality standards. SQA generally works on one or more industry standards that help in building software quality guidelines and implementation strategies.

It includes the following activities −

* Process definition and implementation
* Auditing
* Training

SQA Activities

***Given below is the list of SQA activities:***

**#1) Creating an SQA Management Plan:**

The foremost activity includes laying down a proper plan regarding how the SQA will be carried out in your project. Along with what SQA approach you are going to follow, what engineering activities will be carried out, and it includes ensuring that you have a right talent mix in your team.

**#2) Setting the Checkpoints:**

The SQA team sets up different checkpoints according to which it evaluates the quality of the project activities at each checkpoint/project stage. This ensures regular quality inspection and working as per the schedule.

**#3) Apply software Engineering Techniques:**

Applying some software engineering techniques aids a software designer in achieving high-quality specification. For gathering information, a designer may use techniques such as interviews and FAST (Functional Analysis System Technique).

**#4) Executing Formal Technical Reviews:**

An FTR is done to evaluate the quality and design of the prototype. In this process, a meeting is conducted with the technical staff to discuss regarding the actual quality requirements of the software and the design quality of the prototype. This activity helps in detecting errors in the early phase of SDLC and reduces rework effort in the later phases.

**#5) Having a Multi- Testing Strategy:**

By multi-testing strategy, we mean that one should not rely on any single testing approach, instead, multiple types of testing should be performed so that the software product can be tested well from all angles to ensure better quality.

**#6) Enforcing Process Adherence:**

This activity insists the need for process adherence during the software development process. The development process should also stick to the defined procedures.

**This activity is a blend of two sub-activities which are explained below in detail:**

**(i) Product Evaluation:**

This activity confirms that the software product is meeting the requirements that were discovered in the project management plan. It ensures that the set standards for the project are followed correctly.

**(ii) Process Monitoring:**

This activity verifies if the correct steps were taken during software development. This is done by matching the actually taken steps against the documented steps.

**#7) Controlling Change:**

In this activity, we use a mix of manual procedures and automated tools to have a mechanism for change control. By validating the change requests, evaluating the nature of change and controlling the change effect, it is ensured that the software quality is maintained during the development and maintenance phases.

**#8) Measure Change Impact:**

If the QA team reports any defect, then the concerned team fixes the defect. After this, the QA team should determine the impact of the change, which is brought by this defect fix. They need to test not only if the change has fixed the defect, but also if the change is compatible with the whole project. For this purpose, we use software quality metrics, which allows managers and developers to observe the activities and proposed changes from the beginning till the end of SDLC and initiate corrective action wherever required.

**#9) Performing SQA Audits:**

The SQA audit inspects the entire actual SDLC process followed by comparing it against the established process. It also checks whatever reported by the team in the status reports were actually performed or not. This activity also exposes any non-compliance issues.

**#10) Maintaining Records and Reports:**

It is crucial to keep the necessary documentation related to SQA and share the required SQA information with the stakeholders. The test results, audit results, review reports, change requests documentation, etc. should be kept for future reference.

**#11) Manage Good Relations:**

In fact, it is very important to maintain harmony between the QA and the development team. We often hear that testers and developers often feel superior to each other. This should be avoided as it can affect the overall project quality.

## Testing

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. Testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

## Applications of Software Testing

* **Cost Effective Development** - Early testing saves both time and cost in many aspects, however reducing the cost without testing may result in improper design of a software application rendering the product useless.
* **Product Improvement** - During the SDLC phases, testing is never a time-consuming process. However diagnosing and fixing the errors identified during proper testing is a time-consuming but productive activity.
* **Test Automation** - Test Automation reduces the testing time, but it is not possible to start test automation at any time during software development. Test automaton should be started when the software has been manually tested and is stable to some extent. Moreover, test automation can never be used if requirements keep changing.
* **Quality Check**- Software testing helps in determining following set of properties of any software such as
  + Functionality
  + Reliability
  + Usability
  + Efficiency
  + Maintainability
  + Portability

## What is Black box Testing?

Black-box testing is a method of software testing that examines the functionality of an application based on the specifications. It is also known as Specifications based testing. Independent Testing Team usually performs this type of testing during the software-testing life cycle. This method of test can be applied to every level of software testing such as unit, integration, system and acceptance testing.

## Behavioural Testing Techniques:

There are different techniques involved in Black Box testing.

* Equivalence Class
* Boundary Value Analysis
* Domain Tests
* Orthogonal Arrays
* Decision Tables
* State Models
* Exploratory Testing
* All-pairs testing

## What is White Box Testing?

White box testing is a testing technique, that examines the program structure and derives test data from the program logic/code. The other names of glass box testing are clear box testing, open box testing, logic driven testing or path driven testing or structural testing.

## White Box Testing Techniques:

* **Statement Coverage -** This technique is aimed at exercising all programming statements with minimal tests.
* **Branch Coverage -**This technique is running a series of tests to ensure that all branches are tested at least once.
* **Path Coverage -**This technique corresponds to testing all possible paths which means that each statement and branch is covered.