**Assignment 2 Student\_id:- 201912043**

**Different types of Metrics:**

1. **Requirements Metrics :-**

Many of errors are caused due to changes in the requirements. In order to eliminate

errors there should be some measurement.

**Uniqueness:** The uniqueness of the metrics can be used to obtain percentage

of the requirement that have been uniquely explained by all reviewers.

=Ri/Rt

Where ,Ri=Requirement with distinctive explanation

Rt=total number of requirements.

Note: if it is less than 95&lt; percentage than you have to do rework.

**Correctness:** this calculate the percentage that all requirement are

correctively validated.

Rc/Rt

Where, Rc=requirement having same interpretation

Rt=total number of requirements.

Note: It must be &gt;=80%

Same as we have to measure Changed requirements, Misinterpreted

requirements,

Understandable requirement, Modifiable requirement, Traced , requirement

testing.

**At the end we have to measure SRS Quality:**

(Rt-(Ref + Red + Rem))/Rt

Where, Rt=requirement tested,

Ref=Error found in SRS

Red= Errors Deleted from SRS

Rem=Error Modified in SRS

Rt=total number of requirement

**2. Designing:**

Various design metrics such as Architectural design metrics, component-level design metrics, user-interface design metrics, and metrics for object-oriented designer used to indicate the complexity, quality, and so on of the software design.

**3. Metrics For Coding:**

First analytic laws for computer science by using a set of primitive measures,

which can be derived once the design phase is complete and code is

generated. These measures are listed below.

n l  = number of distinct operators in a program

n 2  = number of distinct operands in a program

N 1  = total number of operators

N 2 = total number of operands.

By using these measures, Halstead developed an expression for overall

program length, program volume, program difficulty, development effort,

and so on.

Program length (N) can be calculated by using the following equation.

N = n 1 log 2 n l  + n 2  log 2 n 2 .

Program volume (V) can be calculated by using the following equation.

V = N log 2  (n 1 +n 2 ).

Note that program volume depends on the programming language used and

represents the volume of information (in bits) required to specify a program.

Volume ratio (L)can be calculated by using the following equation.

L = Volume of the most compact form of a program

Volume of the actual program

Where, value of L must be less than 1. Volume ratio can also be calculated by

using the following equation.

L = (2/n 1 )\* (n 2 /N 2 ).

Program difficulty level (D) and effort (E)can be calculated by using the

following equations.

D = (n 1 /2)\*(N 2 /n 2 ).

E = D \* V.

**4. Metrics for Testing:-**

It can be varies with different techniques.

Function point can be effectively used to estimate testing effort. Various characteristics like errors discovered, number of test cases needed, testing effort, and so on can be determined by estimating the number of function

points in the current project and comparing them with any previous project.

Metrics used for architectural design can be used to indicate how integration

testing can be carried out.

measures can be used to derive metrics for testing effort. By using program volume (V) and

program level (PL),effort (e)can be calculated by the following equations.

e = V/ PL

Where

PL = 1/ [(n 1 /2) \* (N 2 /n 2 )]     … (1)

For a particular module (z), the percentage of overall testing effort allocated can be

calculated by the following equation.

Percentage of testing effort (z) = e(z)/∑e(i)

For developing metrics for object-oriented (OO) testing, different types of

design metrics that have a direct impact on the testability of object-oriented

system are considered. While developing metrics for OO testing, inheritance

and encapsulation are also considered