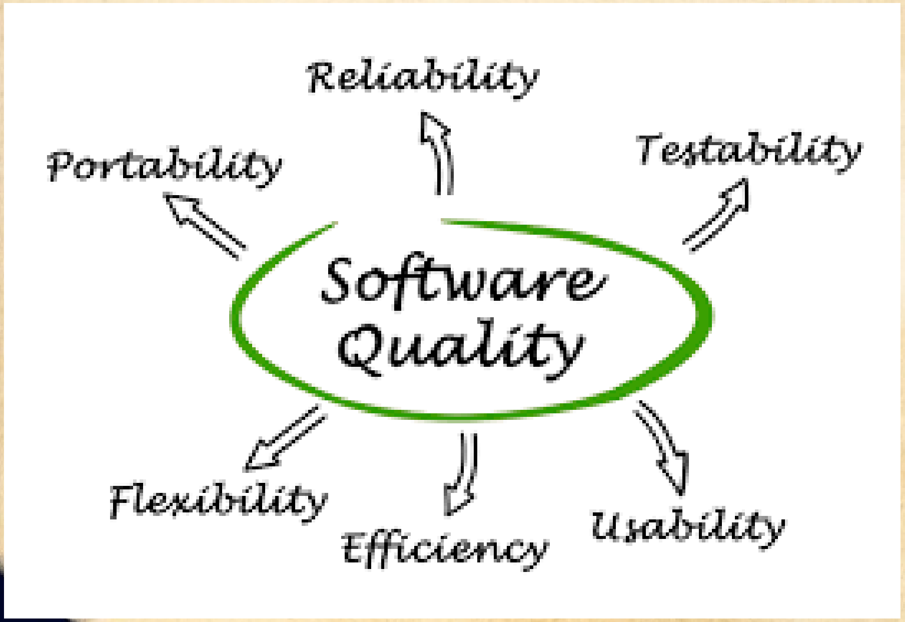
Metrics For Software Quality

What is software quality?

Quality software is reasonably bug or defect free,delivered on time and within budget, meets requirements or expectation , and Is maintainable.



Key aspects of quality for the customer includes:

Good design :looks and style

Good functionality:it does the job well

Reliable:acceptable level of breakdowns or failure consistency

Durable:last as long as it should

Good after sales service

Value for money

Measuring quality

Although there are many measures of software quality
Correctness,maintainability,integrity and usability provide useful indicators for
The project team

Correctness:

- **Is the degree to which the software performs its required function**
- **Defects are those problems reported by user after the programs has been released for general use.**
- **For quality assessment purposes,defects are counted over a standard period of time, typically one year.**
- **The most common measure for correctness is defects per KLOC**

(Defects $\rightarrow$ lack of conformance to requirements)

Maintainability:

- Is the ease with which program can be corrected if an error is encountered,adapted if its environment changes, or enhanced if the customer desires a change in requirements.
- No way to measure directly so we must use indirect measure.
- A simple time-oriented metric is mean-time-to-change(**MTTC**)

  the time it take to analyze change request,design an appropriate modification,implement the change,test it and distribute the change to all users.

Integrity:

- Measures a systems ability to withstand attacks(both accidental and intentional) to its security.
- To measure integrity we should know threat and security.
- Threat is the probability(estimated or derived from empirical evidence)that can attack of a specific type will occur within a given time.
- Security is the probability that the attack a specific type will be repelled.
- Integrity =Σ[1-(threat*(1-security))]

Example:threat=0.25 and security=0.95
Then integrity=0.99(very high)
If,threat=0.50
Security=0.25
Integrity=0.63(unacceptably low)

Usabiliity:
It indicates the usability i.e., the extent to which the user's feel the software as easy to use.

Defect removal Efficiency:

A quality metric that provides benefit at both the project and process level is defect removal efficiency(DRE)

DRE=E/(E+D)

Where E is the number of errors found before delivery of the software to the end user and D is then number of defects found after delivery.

- Ideal value for DRE is 1.(i.e no defects are found in the software)
- As E increases, it is likely that the final value of D will decrease(errors are filtered out before they become defects.
- DRE encourages a software project team to find as many errors as possible before delivery.
- DRE can also be used within the project to assess a teams ability to find errors before they are passed to the next frame work activitiy.
- **DREi=Ei/(Ei+Ei+1)**,where Ei is the number of errors found during software engineering action i and Ei+1 is the number of errors found during software engineering action i+1
- A quality objective for a software team(or an individual software engineer) is to achieve DREi that approaches 1.

Metrics for small organizations

- The vast majority of software development organizations have fewer than 20 software people.
- It is unreasonable, and in most cases unrealistic to expect that such organizations will develop comprehensive software metrics programs.
- However, they must focus on metrics to help improve their local software process, quality and timeliness of the product they produce.
- A small organization can begin by focusing not on measurement but rather on results.
- The software group can define a single objective that requires improvements.
- **For example Reduce the time to evaluate and implement change requests. A small organization might select the following set of easily collected measures.**

- Tqueue :time(hours or day) elapsed from the time request is made until evaluation is completed
- Weval:effort(person-hours)to perform the evaluation
- Teval:time(hours or days)elapsed from completion of evaluation to assignment of change order to personnel.
- Wchange:effort(person-hours) required to make the change.
- Tchange:time required(hours or days) to make the change.
- Echange:errors uncovered during work to make change.
- Dchange:defects uncovered after changes is released to the customer.

Once these measures have been collected for a number of change requests
It is possible to compute the total elapsed time from change request to implementation
Of the change and the percentage of elapsed time absorbed by
Initial queuing,evaluation and change assignment, and change
implementation

$$DRE = Echange/(Echange + Dchange)$$