Software Measurements & Metrics

DR JULIANA JAAFAR

Learning Objective

To introduce the concept of

- Measure, Measurement & Metrics
- Software Quality Metrics

- The objectives of Quality Measurement
- Software Quality Frameworks & IEEE Software Quality Metric Methodology
- Features of Good Quality Metrics
- Limitation of Software Quality Metrics

How big is the system?

How many bugs were found before releasing version 1.0?

How many bugs did the customers find in the first three months after releasing the system?

What was the overall productivity of the software development team in developing the product?

Is the new system has better quality than the previous system?

"You can't control what you can't measure"

(Tom DeMarco, 1982)

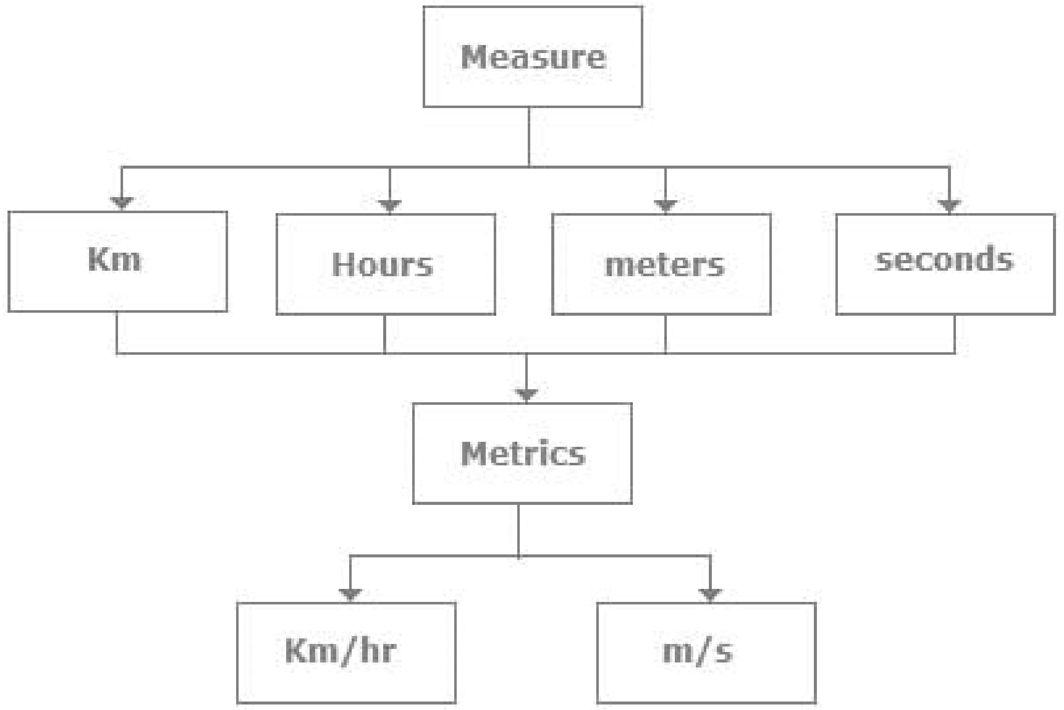
Measurement, Measure, Metrics

- **Measurement** is the act of obtaining the measure

- **Measure** provides quantitative indication of the size of some product or process attribute. E.g. *No of errors*

- **Metrics** is a quantitative measure of the degree to which a system, component, or process possesses a given attribute. E.g. *No of errors found per person hours expended*

Measure Vs Metrics



Definition : Software Quality Metrics

- Measurement of attributes, pertaining to software quality along with its process of development.

IEEE, 1990

(1) **A quantitative measure** of the degree to which an item possesses a given quality attribute.

(2) **A function** whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which the software possesses a given quality attribute.

Objectives of Quality Measurement

1. **To define a clear quality requirements/goals –**

   "Project without clear goals will not achieve their goals clearly" (Tom Gilb, 1988)

2. **To facilitate management control, planning and managerial intervention.**
   - Deviations of actual from planned performance.
   - Deviations of actual timetable and budget performance from planned.

   "You cannot control what you cannot measure"

3. **For corrective action (Prevention) and Process Improvement**
   - Accumulation of metrics information regarding the performance of teams, units, etc.

DEFINE
EVALUATE
PREDICT
IMPROVE

Classification of Software Quality Metrics

Product

Describe the characteristics of product.

E.g. size, complexity, design features, performance, and quality level

PRODUCT
PROCESS
PROJECT

Process

Used for improving software development/maintenance process

E.g. effectiveness of defect removal, pattern of testing defect arrival, and response time of fixes

Project

Describe the project characteristics and execution.

E.g. no of developers, cost, schedule, productivity, etc.

Product Metrics

- Number and type of defects found during requirements, design, code, and test inspections
- Number of pages of documentation delivered
- Number of new source lines of code created
- Number of source lines of code delivered
- Total number or source lines of code delivered
- Average complexity of all modules delivered

Process Metrics

- Average size of modules
- Total number of modules
- Total number of bugs found (by priority and severity) as a result of unit testing and integration testing
- Test coverage ratio
- Number of failures in high importance requirements
- Productivity, as measured by KLOC per person-hour

Process Metrics

- **Help desk (HD) metrics**
  - HD calls density metrics -measured by the number of calls.
  - HD calls severity metrics -the severity of the HD issues raised.
  - HD success metrics –the level of success in responding to HD calls.
- **Corrective maintenance**
  - Software system failures density metrics
  - Software system failures severity metrics
  - Failures of maintenance services metrics
  - Software system availability metrics

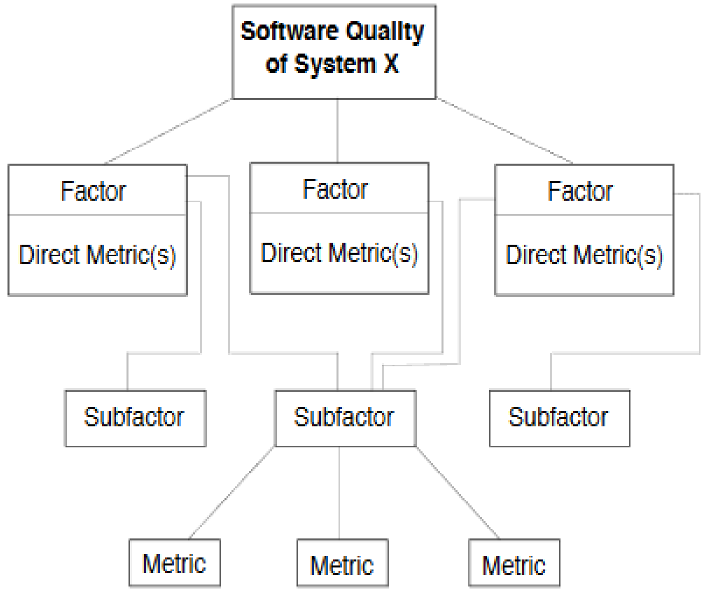
Process Metrics

- Average find-fix cycle time
- Number of person-hours per inspection
- Number of person-hours per KLOC
- Average number of defects found per inspection
- Number of defects found during inspections in each defect category
- Average amount of rework time
- Percentage of modules that were inspected

Software Quality Metrics Frameworks

Objective : To identify software quality metrics

1. Identify **quality requirements** that the software product must meet
2. Define **quality factors** – Management-oriented attributes of software that contribute to its quality
3. Define **quality subfactors** – Decompositions of a quality factor to its technical components
4. Define **metrics to measure the defined quality requirements** – quantitative measures of the degree to which given attributes (factors) are present



Example 1: “Usability”

- Quality requirement : “The product will be easy to use”
- Quality factor(s) : Usability (An attribute that bears on the effort needed for use and on the assessment of such use by users)
- Quality sub-factors : Understandability, ease of learning, operability, communicativeness
- Metrics : <Next Slide>

Example 1 : "Usability Metrics"

- **Metrics:**
  - **Understanding**
    - Learning time: Time for new user to gain basic understanding of features of the software
  - **Ease of learning**
    - Learning time: Time for new user to learn how to perform basic functions of the software
  - **Operability**
    - Operation time: Time required for a user to perform operation(s) of the software
  - **Communicativeness**
    - Human factors: Number of negative comments from new users regarding ergonomics, human factors, etc.

Example 2: “Reliability Metrics”

Production incidents – Number of high priority bugs identified in production per month.

Reliability testing – Load testing – the failure rate under X number of high load, and Regression testing - number of new defects introduced when software undergoes one cycle of changes.

Average failure rate – Average number of failures per period per deployed unit or user of the software.

Mean Time Between Failures (MTBF) and Mean Time To Recover/Repair (MTTR)

Defect Removal Efficiency (DRE)

Example 3: “Performance Metrics”

Load testing – The failure rate under 1,000 concurrent users.

Stress testing - The range of upper limit capacity of the system.

Soak testing – Checking if the system can handle a certain load for a prolonged period of time, and when performance starts to degrade.

Example 4: "Security Metrics"

Time to resolution – Time taken from the time a vulnerability was introduced in the software until a fix or patch is released

Deployment of security updates – For software deployed on users equipment, number of users have actually installed a patch or security update?

Actual security incidents, severity and total time of attacks – Number of times a system actually breached, how badly did the breach affect users, and for how long?

Example 5: “Maintainability and Code Quality Metrics”

Lines of code – A very simple metric that has an impact on the maintainability of a system.
** Software with more lines of code tends to be more difficult to maintain and more prone to code quality issues.

Static code analysis – Automatic examination of code to identify problems and ensure the code adheres to industry standards. Static analysis is done directly on the code without actually executing the software.

Software complexity – Cyclomatic complexity and N-node complexity.
**Code that is more complex is likely to be less maintainable.

Example 6: “Rate of Delivery Metrics”

Number of software releases – This is the basic measurement of how frequently new software is delivered to users.

Agile stories which are “done” in a certain time period – Counting the number of “stories,” or user requirements, which are actually shipped to the user, provides a more granular measure of the rate of delivery.

User consumption of releases – Number of users who download or install a new patch or software update.

Software Quality: The Top 10 Metrics to Build Confidence.

The 10 metrics to rely on when assessing software quality.

By by John Lafleur
Aug. 22, 19 · Agile Zone

1. Number of Bugs — Possibly by Priority or Severity

2. Change Failure Percentage

3. Pull Request Quality

4. Test Coverage Ratio

5. Mean Time Between Failures (MTBF) and Mean Time To Recover/Repair (MTTR)

Source : *https://dzone.com/articles/software-quality-the-top-10-metrics-to-build-confi*

Software Quality: The Top 10 Metrics to Build Confidence.

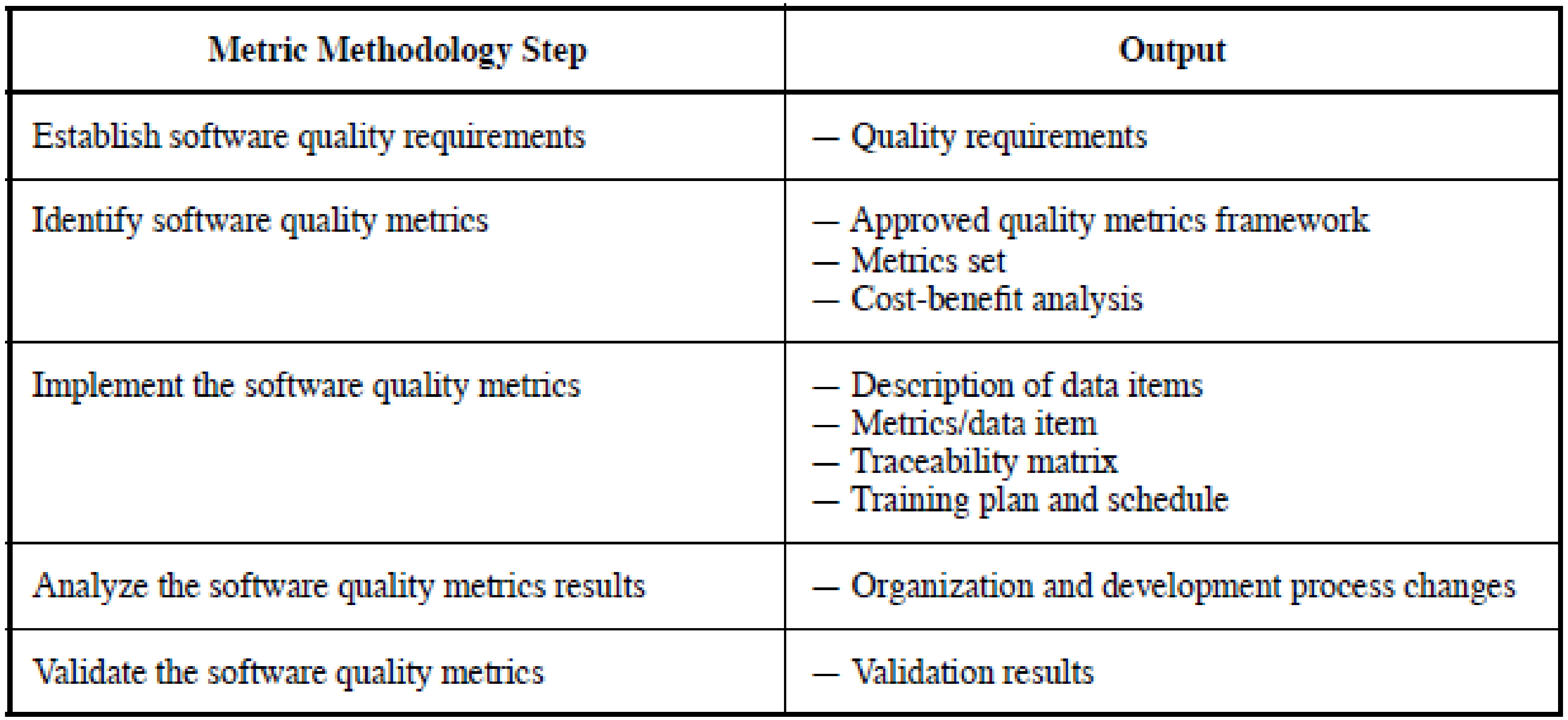
The 10 metrics to rely on when assessing software quality.

By by John Lafleur
Aug. 22, 19 · Agile Zone

6. **Service-Level Agreement (SLA)**
7. **Defect Removal Efficiency (DRE)**
8. **Application Crash Rate (ACR)**
9. **Defect Density**
10. **Age of Dependencies**

Source : *https://dzone.com/articles/software-quality-the-top-10-metrics-to-build-confi*

IEEE Software Quality Metric Methodology



Metric Methodology Step	Output
Establish software quality requirements	— Quality requirements
Identify software quality metrics	— Approved quality metrics framework — Metrics set — Cost-benefit analysis
Implement the software quality metrics	— Description of data items — Metrics/data item — Traceability matrix — Training plan and schedule
Analyze the software quality metrics results	— Organization and development process changes
Validate the software quality metrics	— Validation results

Features of Good Quality Metrics

- **Specific** to measure the particular attribute or an attribute of greater importance.
- **Comprehensive** for wide variety of scenarios.
- Should not consider attributes that have already been measured by some other metric.
- **Reliable** to work similarly in all conditions.
- **Easy** and **simple** to understand and operate.

Limitation of Software Metrics

- **Budget** constraints in allocating the necessary resources (manpower, funds, etc.) for development of a quality metrics system and its regular application.
- **Human factors**, especially opposition of employees to evaluation of their activities.
- **Uncertainty regarding the data's validity**, rooted in partial and biased reporting.

Exercise 1 : Identify a software quality metric from the list of statements

	YES	NO
Project quality plan		
Number of errors per 1000 line of code (KLOC)		
Contract proposal review		
Time required to understand employee payroll calculation module.		
Detailed design inspection		
Test plan sign-off		
Number of severe errors found in software installation plan		

Exercise 1 : Identify a software quality metric from the list of statements

	YES	NO
Room temperature		
Total failure time of hotel tracking system		
Number of changes made to requirements document		

Exercise 2 : Construct the software quality metrics for the following.

- Measure the speed of a student course registration module
- Measure how easy it is to learn new student data entry module
- Measure how many student can be registered in one hour
- Measure the quality of a programmer coding
- Measure the quality of software development plan
- Measure the quality of user manual
- Measure the quality of requirements document

See you NEXT Class..