Scope of SM

Definitions

- *Measure* - quantitative indication of extent, amount, dimension, capacity, or size of some attribute of a product or process.
  - E.g., Number of errors

- *Metric* - quantitative measure of degree to which a system, component or process possesses a given attribute. "A handle or guess about a given attribute."
  - E.g., Number of errors found per person hours expended

Why Measure Software?

- Determine the quality of the current product or process
- Predict qualities of a product/process
- Improve quality of a product/process

Scope of SM

- Cost and Effort Estimation
- Productivity measures and models
- Data collection
- Quality models and measures
- Reliability models
- Performance evaluation and models
- Structural and Complexity metrics
- Capability maturity assessment
- Management by metrics
- Evaluation of methods and tools

Scope of Software Metrics

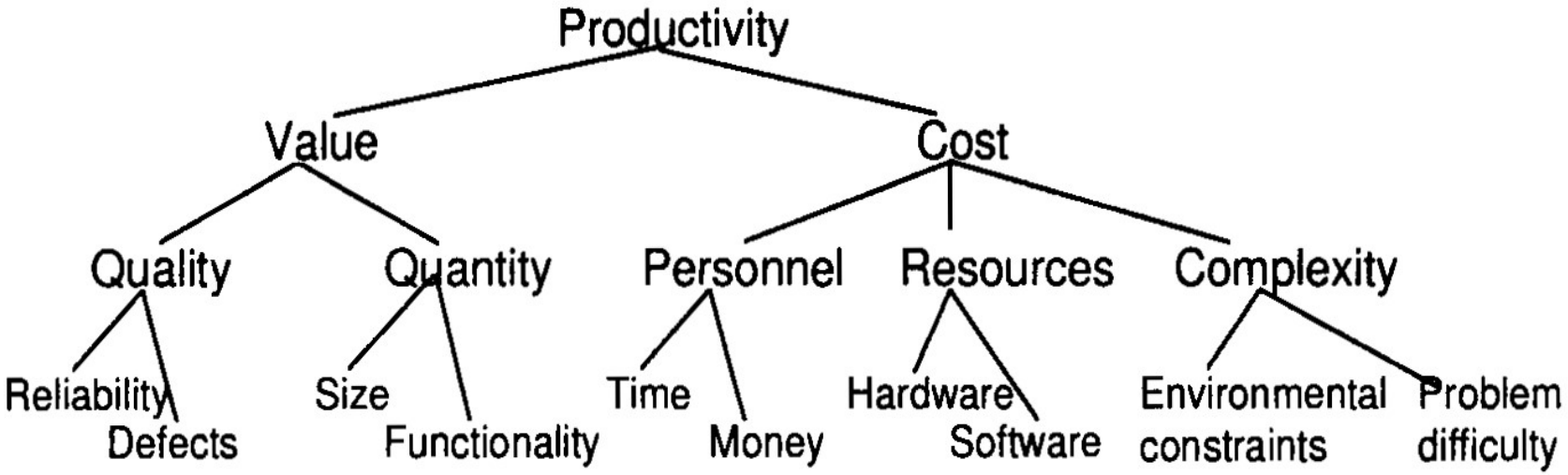
Cost and effort estimation

- Software cost estimation is the process of predicting the amount of effort required to build a software system.
- Estimates for project cost and time requirements are derived during the planning stage of a project.
- Models used to estimate cost can be categorized as either cost models(e.g., Constructive Cost Model COCOMO).
- Experience is often the only guide used to derive these estimates, but it may be insufficient if the project breaks new ground.

Scope of Software Metrics

Productivity models and measures

- Definition: The rate of output per unit of input.
  - Productivity = size/effort
  - Productivity = LOC/person-month
- Productivity model based on decomposition to measurable attributes:



Scope of Software Metrics

Data collection

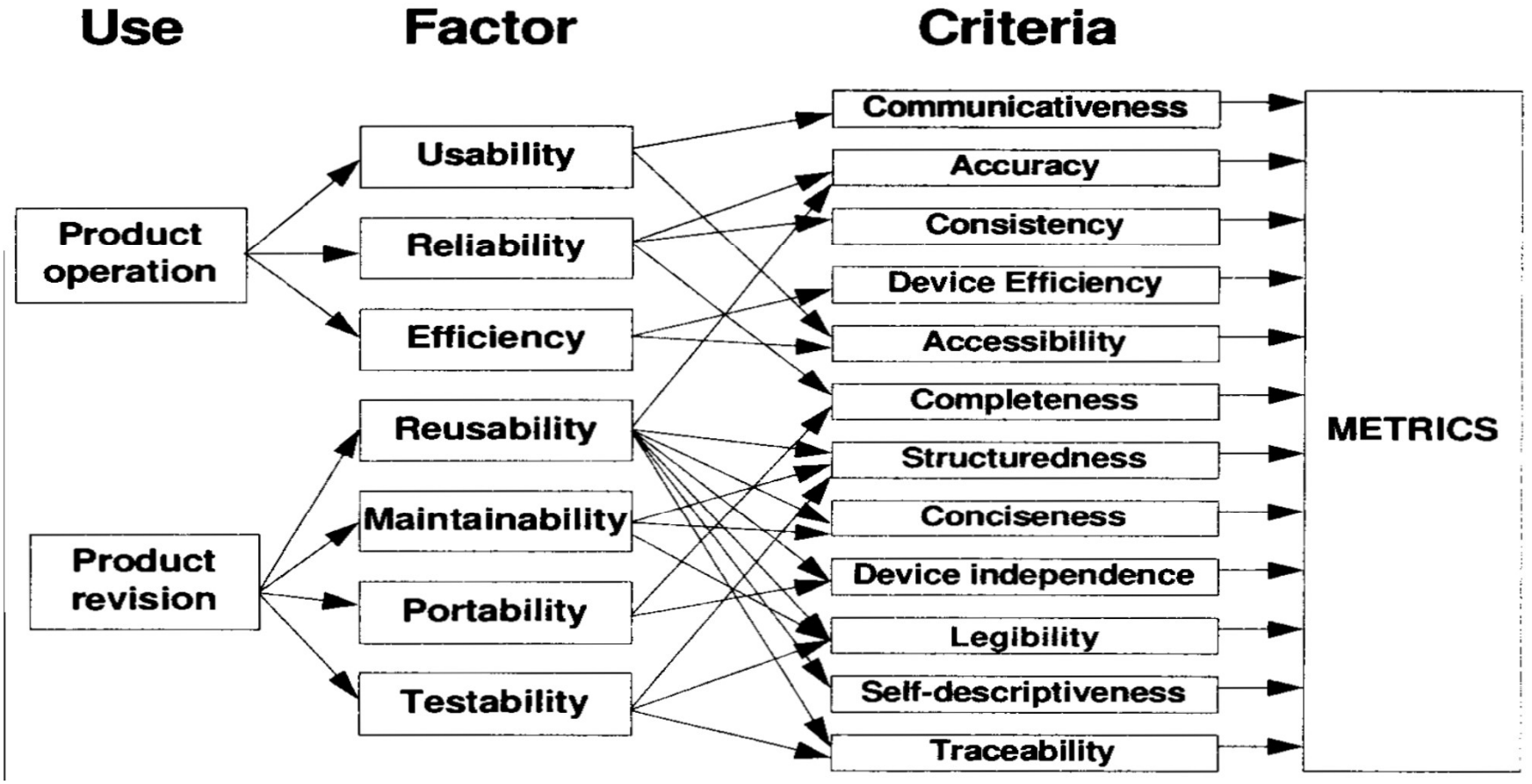
- What data should be collected?
- How it should be collected?
- Is collected data reproducible?

Example : software failure data collection

- Time of failure
- Time interval between failures
- Cumulative failure up to a given time
- Failures experienced in a time interval.

Scope of Software Metrics

Quality models and measures



Scope of Software Metrics

Reliability models

- Plot the change of failure intensity() against time.
- Many models are proposed like basic exponential model and logarithmic Poisson model.
- The basic exponential model assumes finite failures in infinite time; the logarithmic Poisson model assumes infinite failures.
- Some Automated tools are available.

Scope of Software Metrics

Performance evaluation and models

- Using externally observable performance characteristics such as response time and completion rate
- Efficiency of algorithm

Scope of Software Metrics

Structural and complexity metrics

- Control flow structure
- Data-flow structure
- Data structure
- Information flow attributes

- Complexity metrics
  - Cyclomatic complexity (McCabe 1989) defining number of independent paths in execution of program.

Scope of Software Metrics

Management by metrics

- Metrics for project control
  - Metrics relates to specification quality
  - Metrics for the design model
  - Metrics for documentation
  - Checking and testing metrics
  - Resource metrics
  - Change metrics

Scope of Software Metrics

Evaluation of methods and tools

- Efficiency of methods
- Efficiency and reliability of tools
- Certification test of acquired tools and components
- Benchmarking

Scope of Software Metrics

Capability maturity assessment

- US Software Engineering(SEI) Model (1989): CMM grading using five-level scale.
- ISO 9001: Quality systems: models for quality assurance in design /development production,installation and servicing(1991)
- ISO 9000-3 : Guidelines for application of ISO 9001 to the development,supply and maintenance of software(1991).

How to implement?

- The eight steps required to implement a software measurement program are:
  - Document the software development process
  - State the goals
  - Define metrics required to reach goals-GQM
  - Identify data to collect
  - Define data collection procedures
  - Assemble a metrics toolset
  - Create a metrics database
  - Define the feedback mechanism

Who Benefits From Measurement

Managers

- What does each process cost?
- How productive is the staff?
- How good is the code being developed?
- Will the user be satisfied with the product?
- How can we improve?

Engineers

- Are the requirements testable?
- Have we found all the failures?
- Have we met our product or process goals?
- What can we predict about our software product in the future?