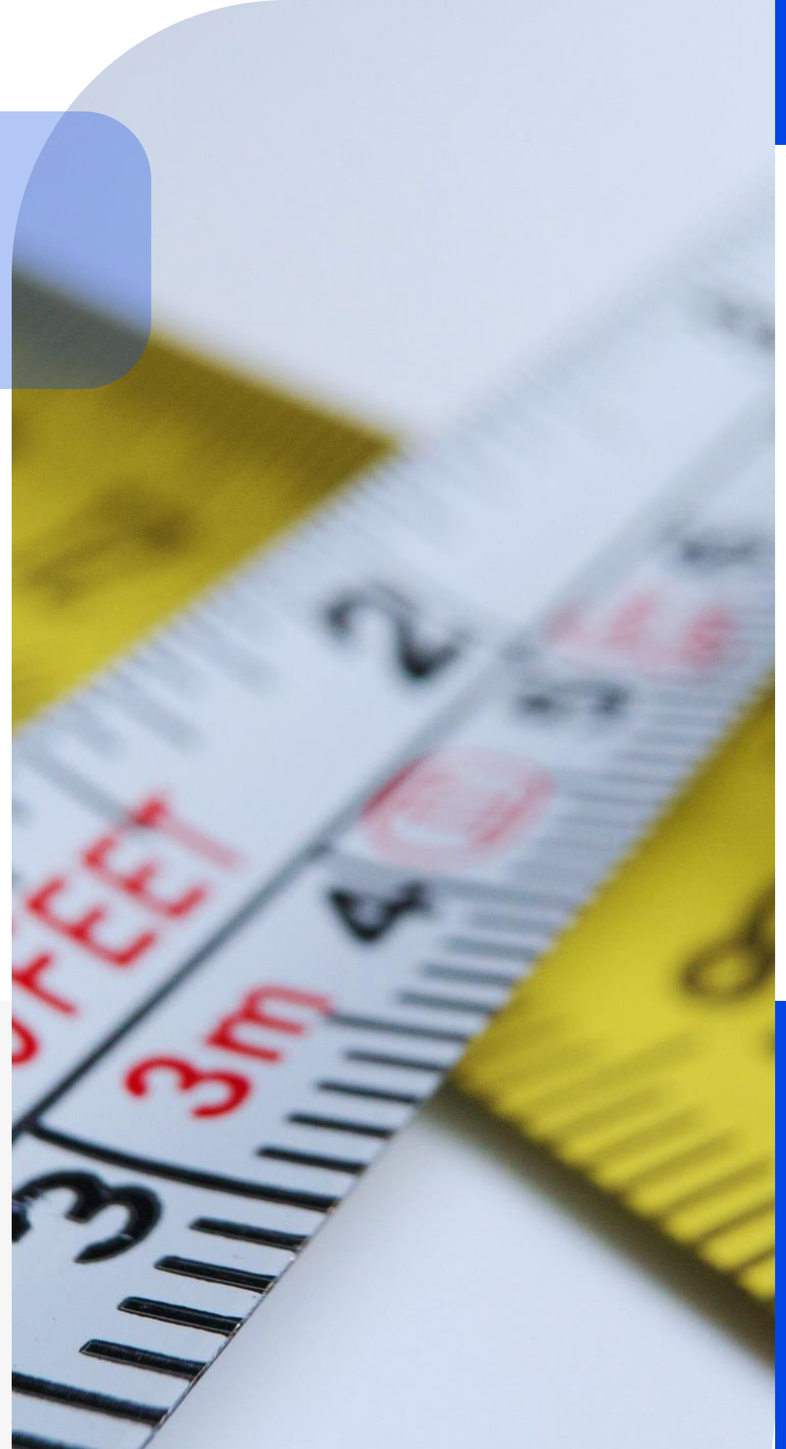
CCS|CSE|CIS1072- Introduction to Software Engineering

SOFTWARE METRICS

08/06/2026



Romenia Silva
romaniasilva@sjp.ac.lk

WHY MEASUREMENT MATTERS

"When you can measure what you are speaking about and express it in numbers, you know something about it; but when you cannot measure, when you cannot express it in numbers, your knowledge is of a meager and unsatisfactory kind."

Sir William Thompson, Lord Kelvin (1824-1907)

MEASURES, METRICS & INDICATORS

MEASURE

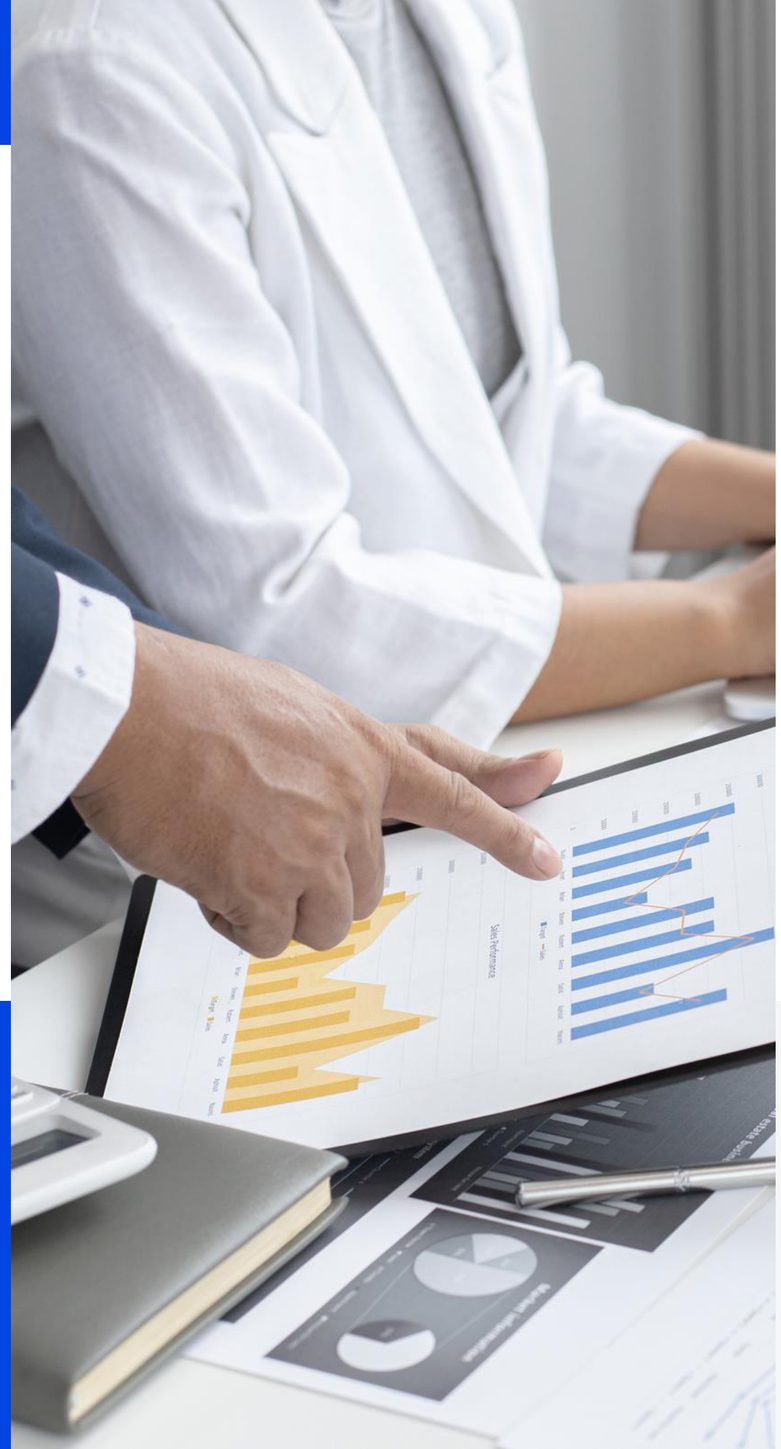
- Quantitative indication of extent, amount, dimension, capacity, or size
- Raw data about specific attributes
- *Examples: Lines of code, number of defects, execution time*

METRIC

- IEEE: "Quantitative measure of the degree to which a system possesses a given attribute"
- Interpreted data with meaning
- *Examples: Defect density, code coverage percentage*

INDICATOR

- A metric or combination of metrics that provide insight into the software process, project, or product
- Actionable intelligence for decision-making
- *Examples: Quality trends, productivity indicators*



EXAMPLE

01 Raw Data:
- 50 bugs found
- 10,000 lines of code

02 Measure:
- Number of defects = 50
- Size of codebase = 10,000 LOC

03 Metric:

Defect Density = 50 / 10,000 LOC = 5 defects per KLOC

04 Indicator: Defect density is **trending upward over time**

05 Decision: Increase code review frequency

THE NEED FOR SOFTWARE METRICS

Software Developers Need To:

Data-driven development decisions

- **Monitor Quality**
  Regular process measurements of evolving systems
- **Specify Requirements**
  Quality and performance in measurable terms
- **Identify Complexity**
  Spot potentially complex components early
- **Measure Reliability**
  Quantify product dependability
- **Compare Quality**
  Against methods, baselines, and targets

THE NEED FOR SOFTWARE METRICS

Software Project Managers Need To:

Evidence-based project management

- **Measure System Costs**

  Determine realistic pricing

- **Measure Staff Productivity**

  Set company-wide goals

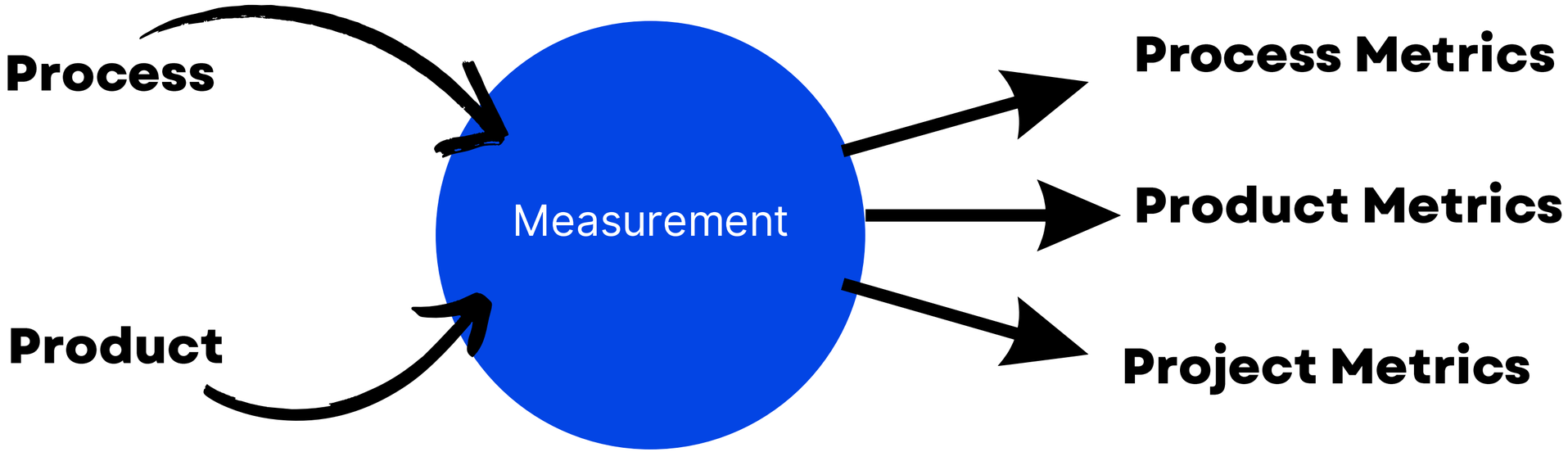
- **Measure Whole-System Costs**

  Better estimate complexity, schedule, resources, budgets

- **Measure Added Value**

  Quantify organizational benefit

A GOOD MANAGER MEASURES



USE OF SOFTWARE METRICS

"You cannot control what you cannot measure"
Tom DeMarco (1982)

Integration with Project Management:

- Form integral part of Project Plan
- Provide data for better control of software production

Three Categories:

1. Product Measurements → Quality, size, complexity
2. Process Measurements → Efficiency, defect rates
3. Project Measurements → Schedule, resources

SUMMARY OF MAIN BENEFITS

VISIBILITY

Improving software process through measurement, recording, and visibility

QUALITY

Assessing software product quality objectively

PRODUCTIVITY

Assessing productivity and improving project estimation

PLANNING & CONTROL

Estimating and controlling projects and their quality

Match the Definitions

Terms	Definitions
1. Measure	A. Provides insight for decision-making
2. Metric	B. Raw quantitative data about an attribute
3. Indicator	C. Interpreted measure with meaning

True or False

1."The system should be fast" is a measurable requirement

2. Developers need metrics mainly for monitoring quality

3. You can control what you cannot measure

4. Software metrics help improve visibility

You found 20 bugs in 5,000 lines of code.

- What's the **measure**?
- What's the **metric**?
- Is 4 bugs/KLOC good or bad? (What do you need to know?)

SCOPE OF SOFTWARE METRICS

Software Metrics has two main fields of application

Measure/assessment of a software product or process

Prediction of critical features involved in a software project (i.e. a future estimate)

SCOPE OF SOFTWARE METRICS

Product Metrics

TYPICAL MEASURES INCLUDE

- Quality of the product and deliverables
- Size and complexity of the product
- Functionality of the product

SCOPE OF SOFTWARE METRICS

Process Metrics

TYPICAL MEASURES INCLUDE

- Productivity
- Statistical SQA - error categorization & analysis
- Defect removal efficiency
- Reuse - components produced and their degree of reusability

TYPICAL SIZE-ORIENTED METRICS

- errors per KLOC (thousand lines of code)
- defects per KLOC
- Rs/= per LOC
- pages of documentation per KLOC
- errors per person-month
- errors per review hour
- LOC per person-month
- Rs/= per page of documentation

FUNCTION POINT ANALYSIS (FPA)

is an ISO recognized method to measure the functional size of an information system. The functional size reflects the amount of functionality that is relevant to and recognized by the user in the business. It is independent of the technology used to implement the system.

FP = Function Points

is a unit of measurement to express the amount of business functionality an information system provides to a user

TYPICAL FUNCTION-ORIENTED METRICS

- errors per FP (thousand lines of code)
- defects per FP
- Rs/= per FP
- pages of documentation per FP
- FP per person-month

EXAMPLE PROCESS QUALITY METRIC

Defect Removal Efficiency (DRE)

$$DRE = \frac{E}{(E + D)}$$

E is the number of errors found before delivery of the software to the end-user

D is the number of defects found after delivery.

SCOPE OF SOFTWARE METRICS

Project Metrics

- used to minimize the development schedule
- used to assess product quality and, when necessary, modify the technical approach to improve quality
- every project should measure

Inputs–measures of the resources (e.g., people, tools)

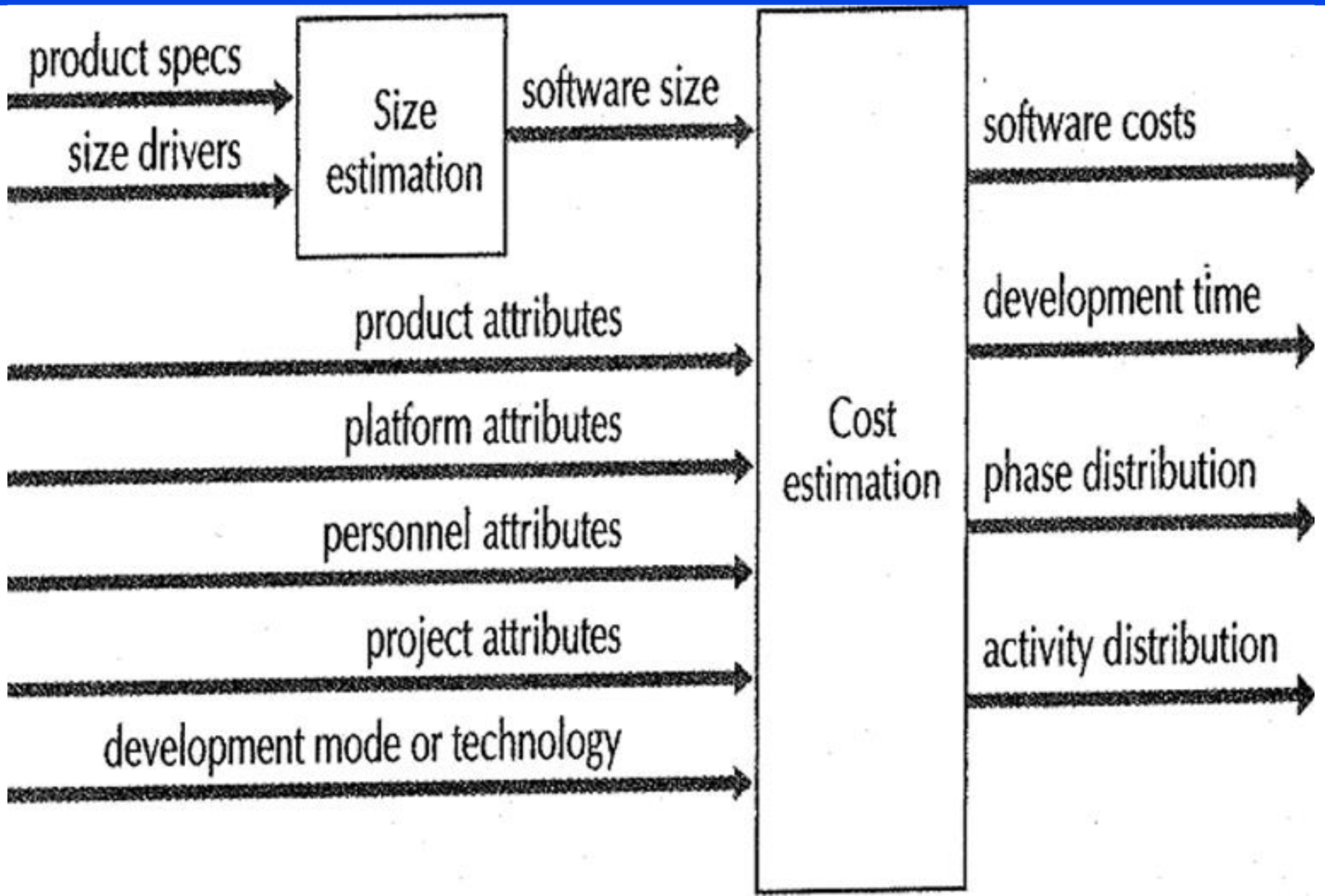
Outputs–measures of the deliverables or work products

Results–measures that indicate the effectiveness of the deliverables.

TYPICAL PROJECT METRICS

- Effort/time per software engineering task
- Errors uncovered per review hour
- Scheduled vs. actual milestone dates
- Changes (number) and their characteristics
- Distribution of effort on software engineering tasks

COST ESTIMATION PROCESS



THANK YOU