Software Metrics

SE 611

Outline of The Course

❏ **Module 1:** Measurement theory

❏ **Module 2:** Software product and process measurements

❏ **Module 3:** Measurement management

Measurement Theory

❏ The basics of measurement

❏ Overview of software measurement metrics

❏ Framework for software measurement

❏ Empirical investigation

❏ Analyzing software measurement data

Software Measurement

❑Measuring Internal Product Attributes: Size and Structure

❑Measuring External Product Attributes

❑Making Process Predictions: estimate effort, size, release date

❑Software Reliability: Measurement and Prediction

Measurement Management

❑Planning measurement

❑Resource management: productivity, teams, and tool

❑Support for measurement

End of This Course…

❏ What is software measurement about?

❏ Why software measurement is important

❏ What does empirical investigation mean in the SE context

❏ What is software measurement metrics

❏ What is software measurement process

❏ How to implement a software measurement plan

❏ Challenges and difficulties of applying software metrics

Chapter 1: Measurement

❑ **Measurement: What Is It and Why Do It?**

❑ **What is Software Metrics**

❑ **Scope of Software Metrics**

Measurement

❑ *Measurement* is the process by which numbers or symbols are assigned to *attributes of entities* in the real world in such a way so as to describe them according to clearly defined rules.

❑ Thus, measurement captures information about *attributes* of *entities*.

❑ An *Entity* is an object or event in the real world

❑ An *attribute* is a feature or property of an *entity*

❑ Two general types of attributes in SE*: Internal (e.g., code, size, and modularity) and External (e.g., reliability, maintainability)*

❑ The accuracy depends on the measuring instrument (or metrics)

Measurement in Software Engineering

❑*Measurement* in SE is selecting, measuring, and putting together many different attributes of the software and adding our subjective interpretation in order to get a whole picture of the software

❑A good software must be reliable, user-friendly, and maintainable

❑So? Is measurements necessary?

❑What are the effect of neglecting proper measuring in SE?

Neglecting of Measurement in SE

❑ Failure to set measurable targets for the product (e.g., how user-friendly, reliable, and maintainable a product is)

❑ Failure to understand and quantify the component costs of a software project, e.g., difference between design cost, coding cost, testing cost

❑ Failure to quantify or predict the quality of product. Thus potential user can not be informed how reliable a product is.

Metrics

- Metrics are standards (i.e., commonly accepted scales, measurements or quantifiable indicators) that define measurable attributes of entities, their units, and their scopes.

- *Software Metrics* are measures that are used to quantify software, software development resources, and/or the software development process

- Common Software metrics: size metrics, effort metrics, quality metrics, productivity metrics, maintainability, and reliability metrics.

- *"What is Not Measurable Make Measurable"*

Mathematical Perspective of Metrics

❑ A metrics is a function *m* defined on pairs of objects *x* and *y* such that $m(x,y)$ represent the distance between *x* and *y*. Such metric must satisfy certain properties

  ❑ $m(x, x) = 0$ for all *x*

  ❑ $m(x, y) = m(y, x)$

  ❑ $m(x, z) <= m(x, y) + m(y, z)$ for all *x*, *y*, and *z*

Software Metrics Challenges

❑ SE Metrics are mostly non-physical

    ❑ *Reliability, maturity, portability, flexibility, maintainability, etc., and relations are unknown*

    ❑ Choosing the right metrics can be challenging.

    ❑ Gathering accurate and reliable data for metrics can be a challenge.

    ❑ Tracking too many metrics can overwhelm teams and make it difficult to focus on the most important aspects.

    ❑ Metrics may not capture all relevant aspects of software development, and they can be influenced by subjective judgments and biases.

    ❑ There is a risk of metric manipulation, where individuals or teams may intentionally or unintentionally manipulate metrics to present a favourable picture

    ❑ Software projects and technologies evolve rapidly, and metrics that were once effective may become outdated or less relevant over time.

Objective of Software Measurement

❑ From manager perspective

    ❑ *How 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?*

❑ From developer perspective

    ❑ *Are the requirements testable?*

    ❑ *Have we found all the faults?*

    ❑ *Have we meet product or process goals?*

    ❑ *What will happen in the future?*

Scope of Software Metrics

❑ Cost and effort estimation models and measures

❑ Data collection

❑ Quality models and measures

❑ Reliability models

❑ Security metrics

❑ Structural and complexity metrics

❑ Capability maturity assessment

❑ Evaluation of methods and tools

Scope of Software Metrics/1

Cost and effort estimation models and measures

- 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) or constraint models (e.g., SLIM).

- Experience is often the only guide used to derive these estimates, but it may be insufficient if the project breaks new ground.

- Many models are available as automated tools.

Scope of Software Metrics/2

❑Data collection

- Very critical and very hard step.
    - What data should be collected?
    - How it should be collected?
    - Is collected data reproducible?
- **Example:** software failure data collection
    1) Time of failure
    2) Time interval between failures
    3) Cumulative failure up to a given time
    4) Failures experienced in a time interval

Scope of Software Metrics/3

❑Quality models and measures
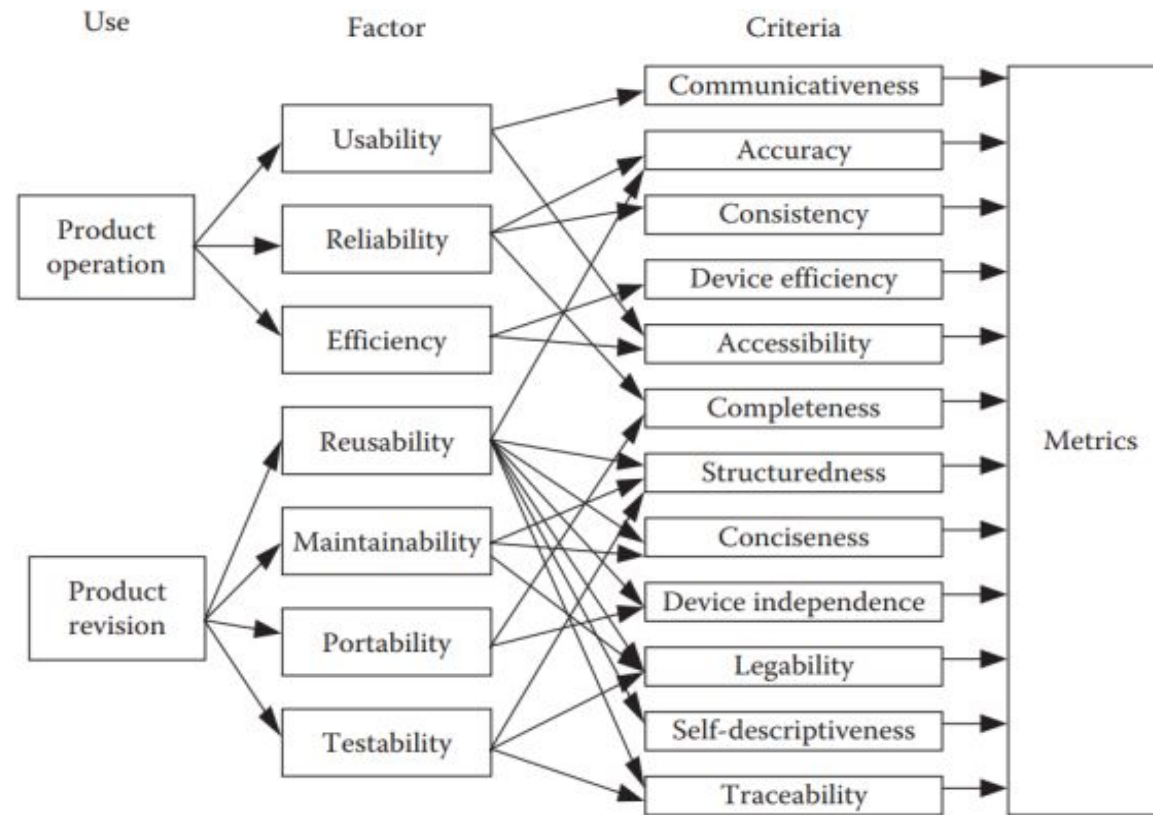
Scope of Software Metrics/4

❑Reliability models

- Plot the change of failure intensity (λ) against time.
- Many models are proposed. The most famous ones are basic exponential model and logarithmic Poisson model.
- The basic exponential model assumes finite failures in infinite time; the logarithmic Poisson model assumes infinite failures.
- Automated tools such as CASRE are available.

Scope of Software Metrics/5

❑Security metrics

    ❑Security depends on both the internal design of a system and the nature of the attacks that originate externally.

    ❑To assess security risks in terms of impact, likelihood, threats, and vulnerabilities.
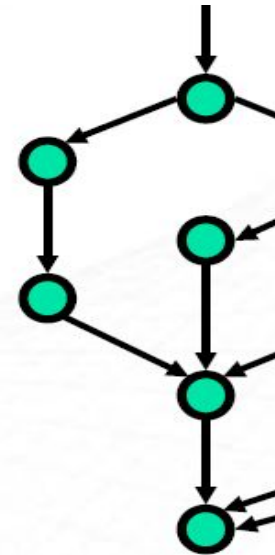
Scope of Software Metrics/6

❑Structural and complexity metrics

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

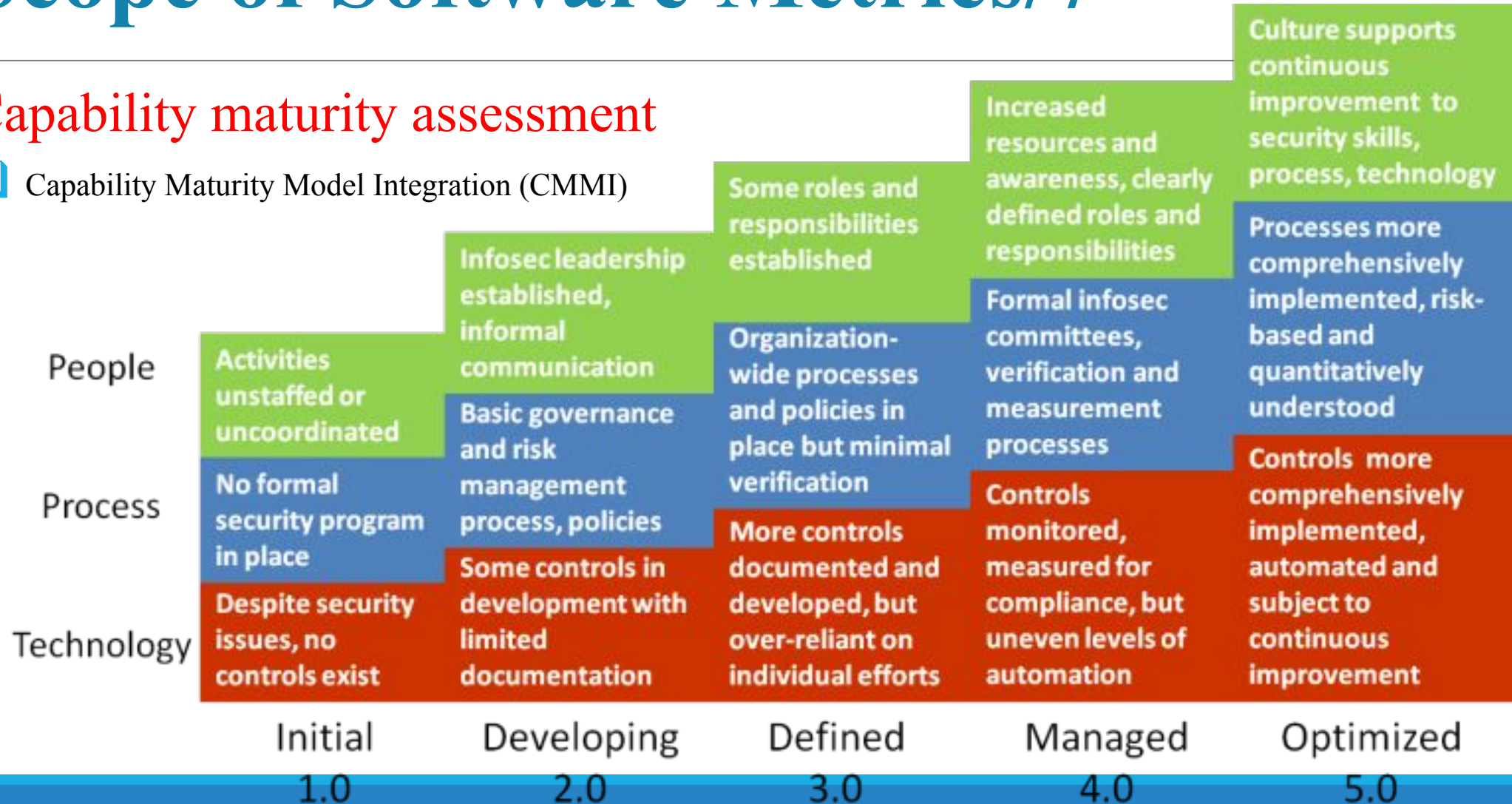Complexity metrics (1979~)

- ■ Cyclomatic complexity (McCabe 1989) defining number of independent paths in execution of a program

Scope of Software Metrics/7

❑ Capability maturity assessment

   ❑ Capability Maturity Model Integration (CMMI)

	Initial 1.0	Developing 2.0	Defined 3.0	Managed 4.0	Optimized 5.0
People	Activities unstaffed or uncoordinated	Infosec leadership established, informal communication	Some roles and responsibilities established	Increased resources and awareness, clearly defined roles and responsibilities	Culture supports continuous improvement to security skills, process, technology
Process	No formal security program in place	Basic governance and risk management process, policies	Organization-wide processes and policies in place but minimal verification	Formal infosec committees, verification and measurement processes	Processes more comprehensively implemented, risk-based and quantitatively understood
Technology	Despite security issues, no controls exist	Some controls in development with limited documentation	More controls documented and developed, but over-reliant on individual efforts	Controls monitored, measured for compliance, but uneven levels of automation	Controls more comprehensively implemented, automated and subject to continuous improvement

Scope of Software Metrics/8

❑Evaluation of methods and tools

- Efficiency of methods (1991~)
- Efficiency and reliability of tools
- Certification test of acquired tools and components
- Benchmarking

Eight Steps of Measurement Program

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

End of Chapter 1