Measurement

SE-843: SOFTWARE METRICS

What Is 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

Measurement captures information about *attributes* of *entities*.

An *entity* is an object (such as a person or a room) or an event (such as the testing phase of a software project) in the real world.

An *attribute* is a feature or property of an entity. Typical attributes include the area or color (of a room), or the elapsed time (of the testing phase).

The accuracy of a measure depends on the measuring instrument as well as on the definition of the measurement. For example, length can be measured accurately as long as the ruler is accurate.

What Is Not Measurable Make Measurable

1. It suggests that one of the aims of science is to find ways to measure attributes of interesting things.
   We should be creating ways to measure our world;
2. Where we can already measure, we should be making our measurements better
3. To improve the rigor of measurement in software engineering, we need not restrict the type or range of measurements
4. Observations that can reduce uncertainty can quantitatively measure the risk of negative events or the likelihood of positive outcomes

Measurement in Software Engineering

Software engineering describes the collection of techniques that apply an engineering approach to the construction and support of software products.

Software engineering activities include managing, costing, planning, modeling, analyzing, specifying, designing, implementing, testing, and maintaining.

However, measurement has been considered a luxury in software engineering !

Neglect of Measurement in Software Engineering

1. We fail to set measurable targets for our software products. For example, we promise that the product will be user-friendly, reliable, and maintainable without specifying clearly and objectively what these terms mean. As a result, from both is complete, we cannot tell if we have met our goals
2. We fail to understand and quantify the component costs of software projects. For example, many projects cannot differentiate the cost of design from the cost of coding or testing. Since excessive cost is a frequent complaint from our customers, we cannot hope to control costs
3. We do not quantify or predict the quality of the products we produce. Thus, we cannot tell a potential user how reliable a product will be in terms of likelihood of failure

Objectives for Software Measurement (*Managers*)

1. *What does each process cost?*
2. *How productive is the staff?*
3. *How good is the code being developed?*
4. *Will the user be satisfied with the product?*

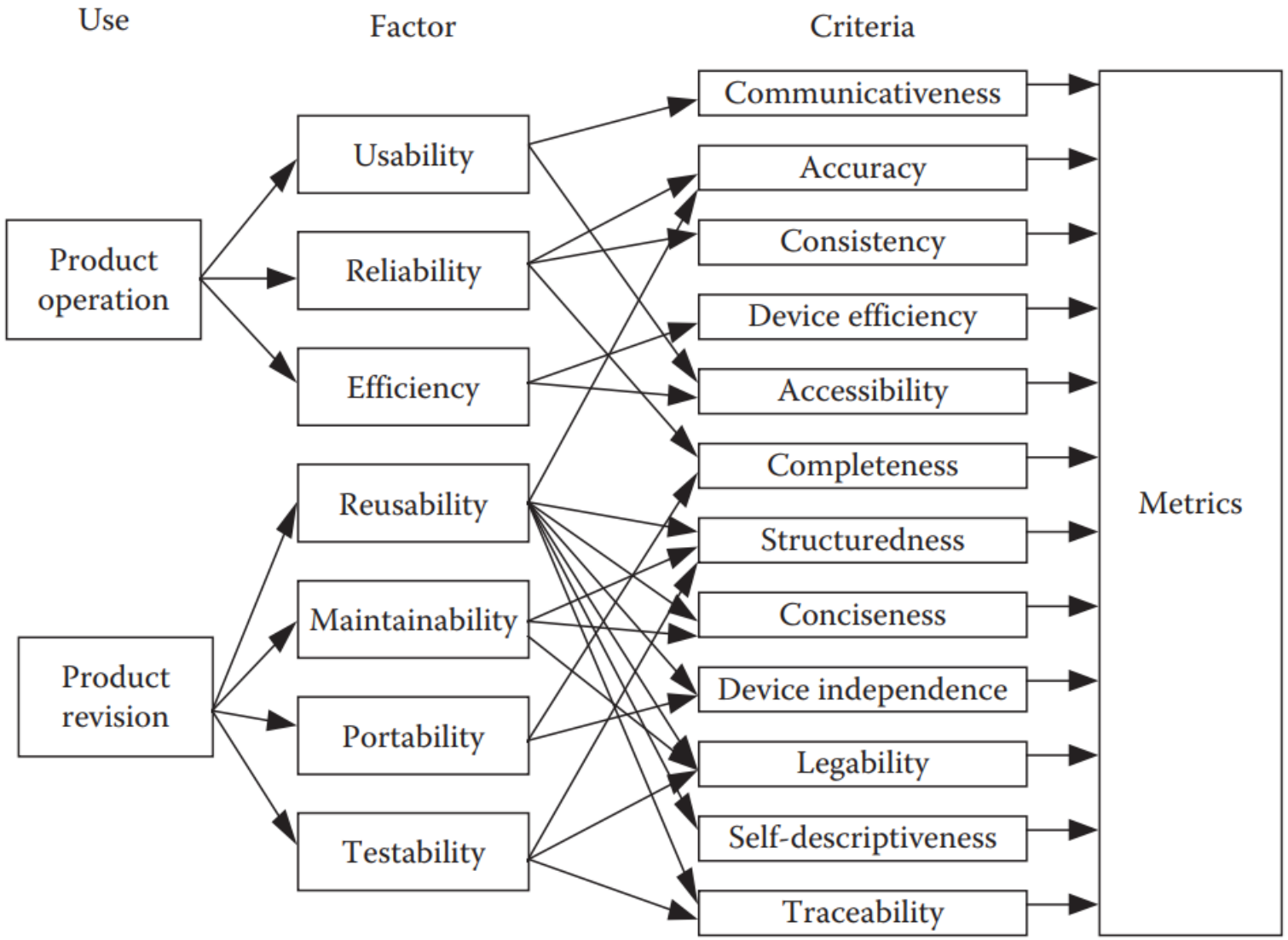
Objectives for Software Measurement (*Developers*)

1. *Are the requirements testable?*
2. *Have we found all the faults?*
3. *Have we met our product or process goals?*
4. *Have we develop all the requirements?*

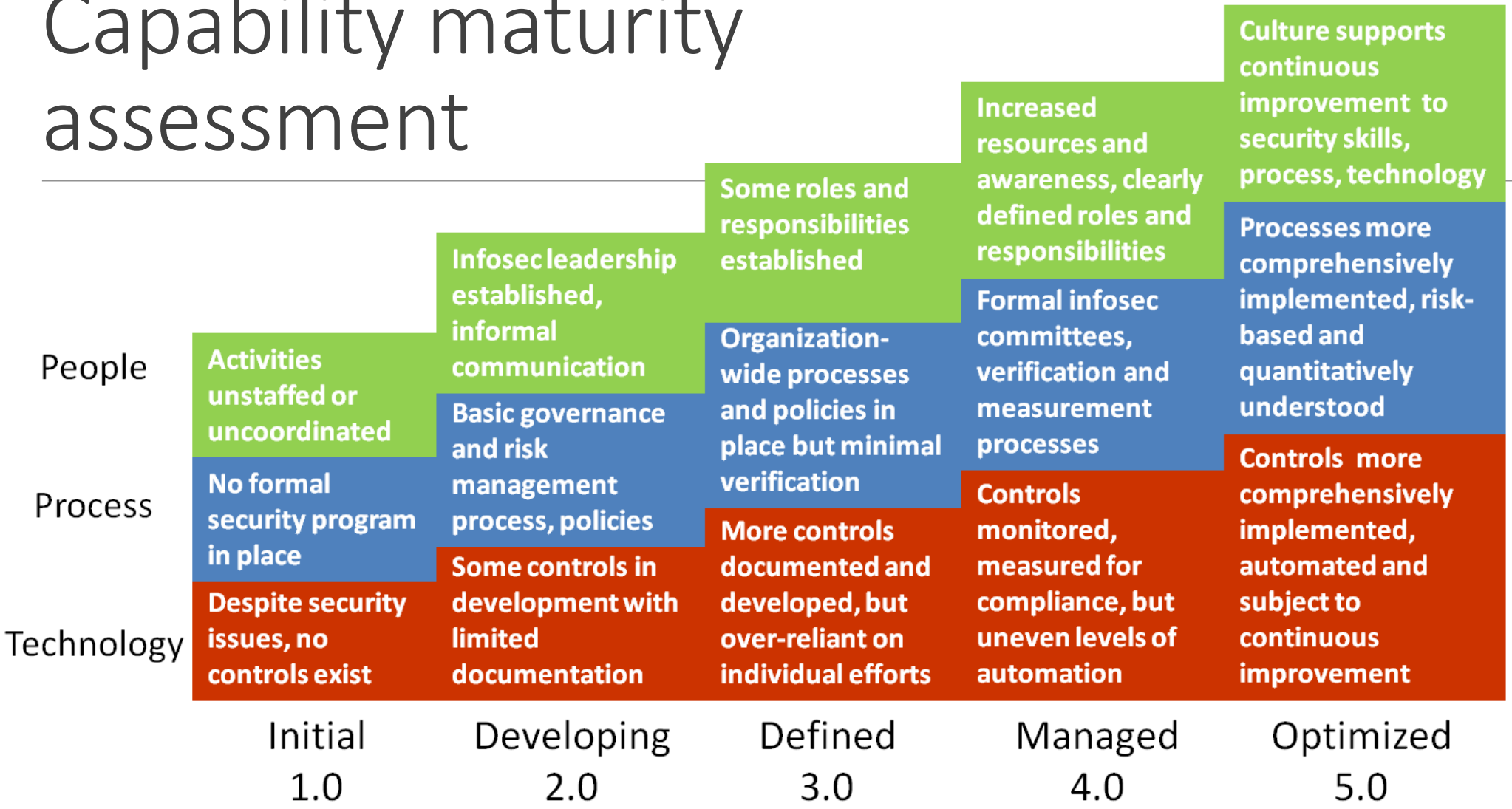
SCOPE OF SOFTWARE METRICS

1. Cost and effort estimation models and measures
2. Data collection
3. Quality models and measures
4. Reliability model
5. Security metrics
6. Structural and complexity metrics
7. Capability maturity assessment
8. Management by metrics
9. Evaluation of methods and tools

Reliability Models



Capability maturity assessment



End of Chapter 1