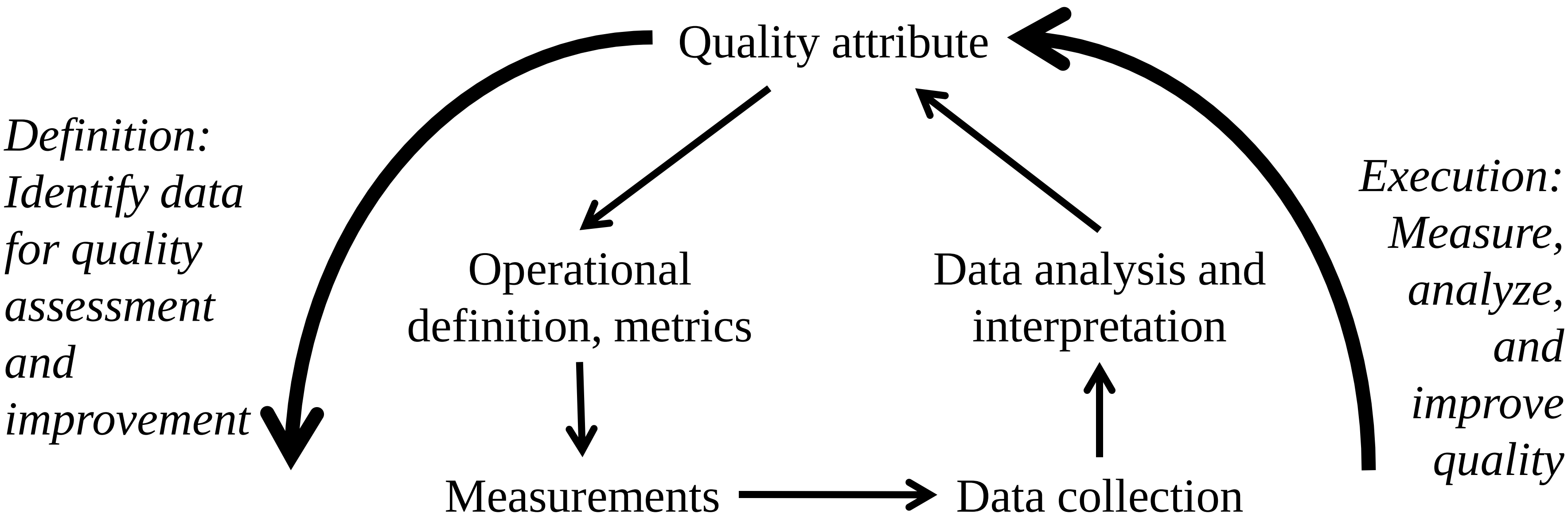
Measurement and Metrics Fundamentals

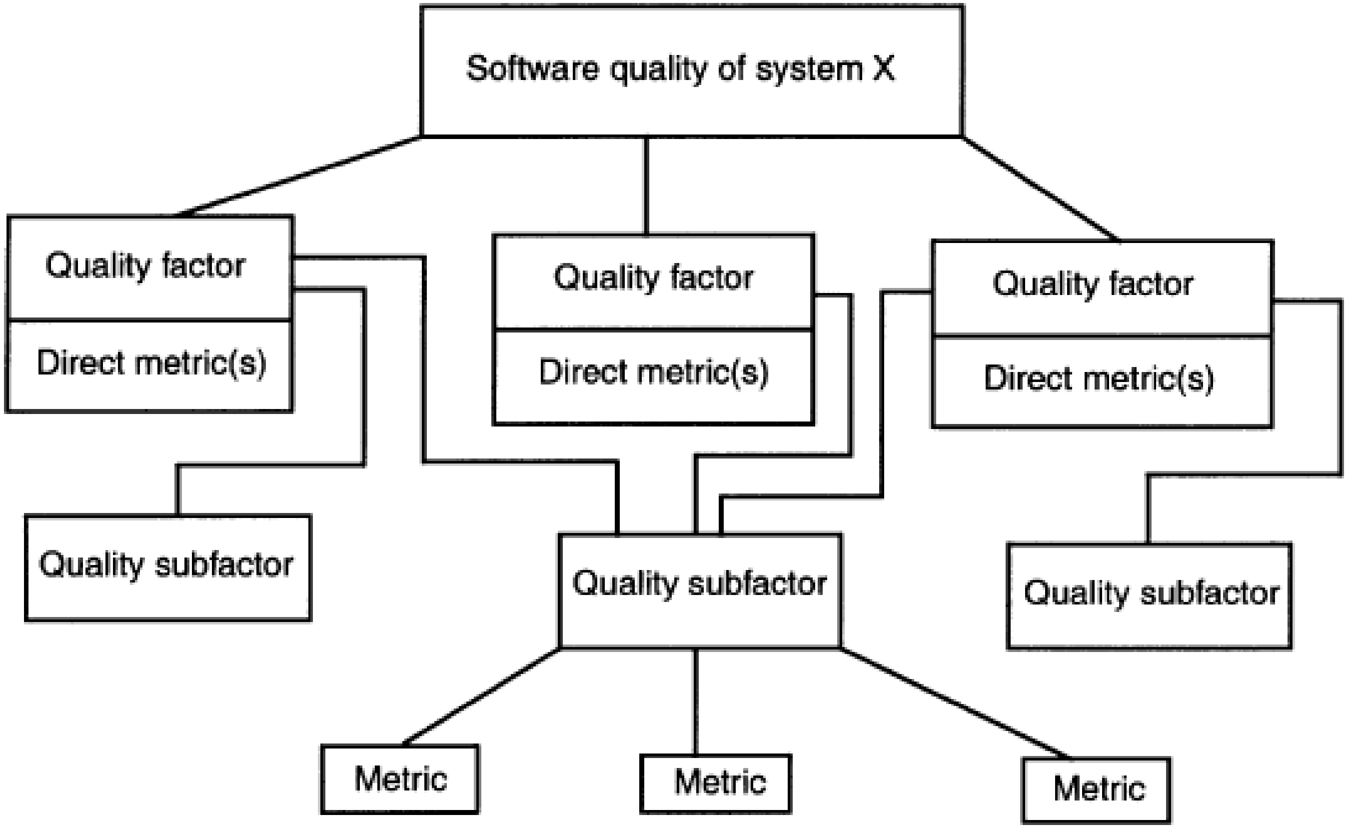
Lecture Objectives

- Provide some basic concepts of metrics
  - Quality attribute ←→ metrics and measurements
  - Reliability, validity, error
  - Correlation and causation
- Discuss process variation and process effectiveness
- Introduce a method for identifying metrics for quality goals
  - Goal-Question-Metric approach

Context: Define Measures and Metrics that are Indicators of Quality



Software Quality Metrics



IEEE-STD-1061-1998(R2004) Standard for Software Quality Metrics Methodology

A Metric Provides Insight on Quality

- A **measure** is a way to ascertain or appraise value by comparing it to a norm [2]
- A **metric** is a quantitative measure of the degree to which a system, component, or process possesses a given attribute [1]
  - **Software quality metric:** A function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which software possesses a given attribute that affects its quality [2]
- An **indicator** is a metric or combination of metrics that provide insight into a process, a project, or the product itself

[1] IEEE-STD-610.12-1990 Glossary of Software Engineering Terminology
[2] IEEE-STD-1061-1998 Standard for Software Quality Metrics Methodology

Measurements vs. Metrics

- A measurement just provides information
  - Example: "Number of defects found during inspection: 12"
- A metric is often derived from one or more measurements or metrics, and provides an assessment (an indicator) of some property of interest:
  - It must facilitate comparisons
    - It must be meaningful across contexts, that is, it has some degree of context independence
  - Example: "Rate of finding defects during the inspection = 8 / hour"
  - Example: "Defect density of the software inspected = 0.2 defects/KLOC"
  - Example:"Inspection effort per defect found = 0.83 hours"

Operational Definition

Concept

↓

Definition

↓

<u>Operational Definition</u>

↓

Measurements

- Concept is what we want to measure, for example, "cycletime"
- We need a definition for this: "elapsed time to do the task"
- The operational definition spells out the procedural details of how exactly the measurement is done
  - "Cycletime is the calendar time between the date when the project initiation document is approved to the date of full market release of the product"

Operational Definition Example

- One operational definition of “development cycletime” is:
  - The cycletime clock starts when effort is first put into project requirements activities (still somewhat vague)
  - The cycletime clock ends on the date of release
  - If development is suspended due to activities beyond a local organization’s control, the cycletime clock will be stopped, and restarted again when development resumes
    - This is decided by the project manager
- Separate “development cycle time” from “project cycletime” which has no clock stoppage and beginning at first customer contact
- The operational definition addresses various issues related to gathering the data, so that data gathering is more consistent

Measurement Scales

- Nominal scale: categorization
  - Different categories, not better or worse
  - Example: Type of risk: business, technical, requirements, etc.
- Ordinal scale: Categories with ordering
  - Example: CMM maturity levels, defect severity
  - Sometimes averages quoted, but only marginally meaningful
- Interval scale: Numeric, but “relative” scale
  - Example: GPAs. Differences more meaningful than ratios
  - “2” is not to be interpreted as twice as much as “1”
- Ratio scale: Numeric scale with “absolute” zero
  - Ratios are meaningful and can be compared

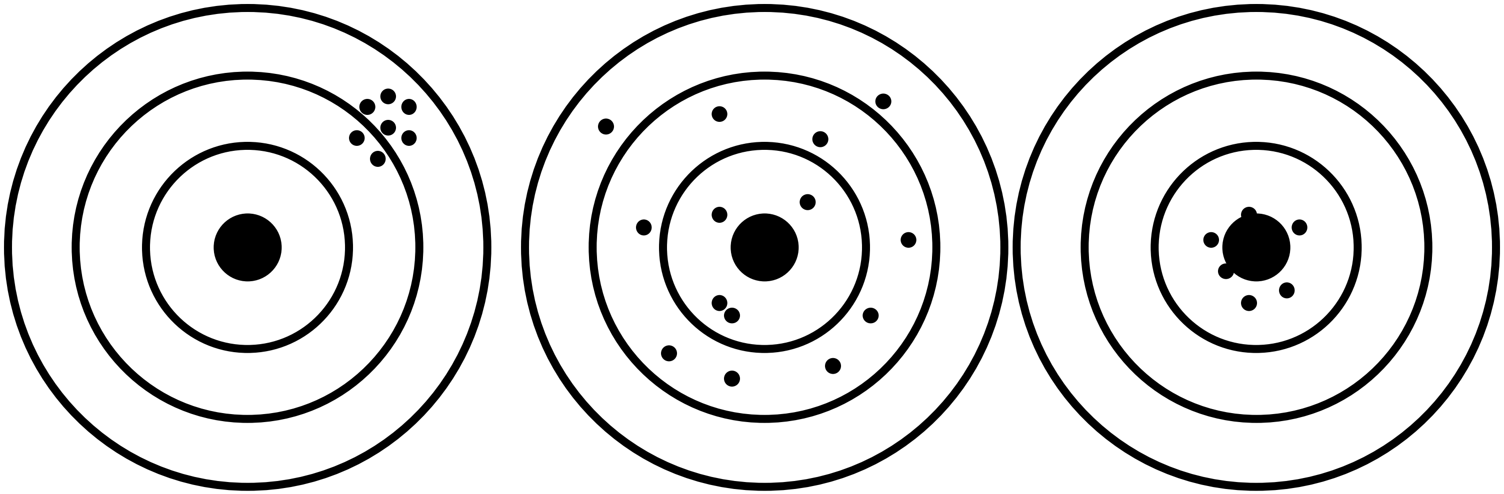
Increasing information content and analysis tools

Using Basic Measures

- See Kan text for good discussion on this material
- Ratios are useful to compare magnitudes
- Proportions (fractions, decimals, percentages) are useful when discussing parts of a whole
  - Such as a pie chart
- When number of cases is small, percentages are often less meaningful – Actual numbers may carry more information
  - Because percentages can shift so dramatically with single instances (high impact of randomness)
- When using rates, better if denominator is relevant to opportunity of occurrence of event
  - Requirements changes per month, or per project, or per page of requirements more meaningful than per staff member

Reliability & Validity

- Reliability is whether measurements are consistent when performed repeatedly
  - Example: Will process maturity assessments produce the "same" outcomes when performed by different people?
  - Example: If we measure repeatedly the reliability of a product, will we get consistent numbers?
- Validity is the extent to which the measurement actually measures what we intend to measure
  - Construct validity: Match between operational definition and the objective
  - Content validity: Does it cover all aspects? (Do we need more measurements?)
  - Predictive validity: How well does the measurement serve to predict whether the objective will be met?



Reliable but not valid | Valid but not reliable | Valid *and* reliable

Figure 3.4, pp. 72 of Kan textbook

Reliable: consistent measurements when using the same measurement method on the same subject

Valid: Whether the metric or measurement really measures or gives insight on the concept or quality attribute that you want to understand

Reliability vs. Validity

- Rigorous operational definitions of how the measurement will be collected can improve reliability, but worsen validity
  - Example: “When does the cycletime clock start?”
- If we allow too much flexibility in data gathering, the results may be more valid, but less reliable
  - Too much dependency on who is gathering the data
- Good measurement systems design often needs a balance between reliability & validity
  - A common error is to focus on what can be gathered reliably (“observable & measurable”), and lose out on validity
  - “We can’t measure this, so I will ignore it”, followed by “The numbers say this, hence it must be true”
    - Example: SAT scores for college admissions decisions
  - Measure what is necessary, not what is easy

Systematic & Random Error

- Gaps in reliability lead to <u>random error</u>
  - Variation between "true value" and "measured value"
- Gaps in validity may lead to <u>systematic error</u>
  - "Biases" that lead to consistent underestimation or overestimation
  - Example: Cycletime clock stops on release date rather than when customer completes acceptance testing
- From a mathematical perspective:
  - We want to minimize the sum of the two error terms, for single measurements to be meaningful
  - Trend information is better if random error is less
  - When we use averages of multiple measurements (such as organizational data), systematic error is more worrisome
    - Broader measurement scope → Broader impact of error

Assessing Reliability

- Can relatively easily check if measurements are highly subject to random variation:
  - Split sample into halves and see if results match
  - Re-test and see if results match
- We can figure out how reliable our results are, and factor that into metrics interpretation
- Can also be used numerically to get better statistical pictures of the data
  - Example: Kan text describes how the reliability measure can be used to correct for attenuation in correlation coefficients (p. 76-77)

Correlation

- Checking for relationships between two variables:
  - Example: Does defect density increase with product size?
  - Plot one against the other and see if there is a pattern
- Statistical techniques to compute correlation coefficients:
  - Most of the time, we only look for linear relationships
  - Text explains the possibility of non-linear relationships, and shows how the curves and data might look
- Common major error: Assuming correlation implies causality (A changes as B changes, hence A causes B)
  - Example: Defect density increases as product size increases → Writing more code increases the chance of coding errors!

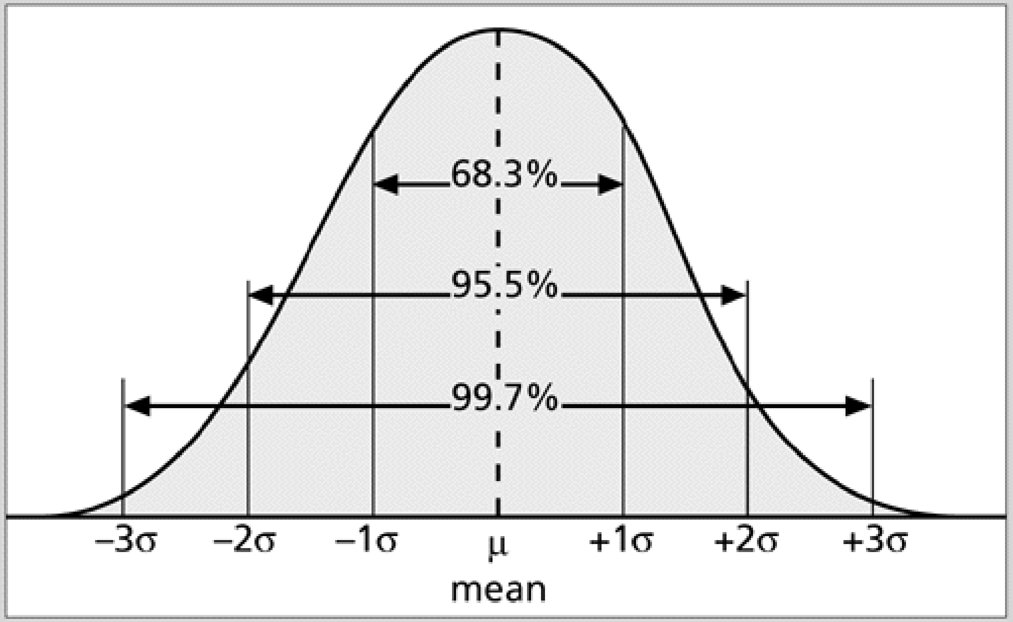
Criteria for Causality

- Observation indicates correlation
- Cause precedes effect in time or logical dependence
- The cause is not spurious
  - Not so easy to figure out! (See diagrams in text p. 81)
  - Maybe common cause for both
    - Example: Code size and defects are a result of problem complexity
  - Maybe there is an intermediate variable
    - Size → number of dependencies → defect rate
    - Why is this important? Because it affects quality management approach
    - For example, we may focus on dependency reduction
  - Maybe both are indicators of something else:
    - Example: developer competence (less competent: more size, defects)

Measuring Process Effectiveness

- A major concern in process theory (particularly in manufacturing) is "reducing process variation"
  - If you are doing the same thing, then do it the same way
    - Monitor the output to make sure that the process is "in control"
  - It is about "improving process effectiveness" so that the process consistently delivers non-defective results
- Process effectiveness is measured as "sigma level"

The Normal Curve



The normal curve

Sigma level is the area under the curve between the limits

- Percentage of situations that are "within tolerable limits"

Six Sigma

- Given "tolerance limits" (the definition of what is defective), if we want +/- 6σ to fit within the limits, the curve must become very narrow:
  - We must "reduce process variation" so that the outcomes are highly consistent
  - Area within +/- 6σ is 99.9999998%
    - ~2 defects per billion
  - This assumes a normal curve. But actual curve is often a "shifted" curve, for which it is a bit different
  - The Motorola (and generally accepted) definition is 3.4 defects per million operations

Why So Stringent?

- Because manufacturing involves thousands of process steps, and output quality is dependent on getting every single one of them right:
  - Need very high reliability at each step to get reasonable probability of end-to-end correctness
  - At 6 sigma, product defect rate is ~10% with ~1200 process steps
  - Concept came originally from chip manufacturing
- Software has sort of the same characteristics:
  - To function correctly, each line has to be correct
  - A common translation is 3.4 defects per million lines of code

Six Sigma Focus

- Six sigma is NOT actually about "achieving the numbers," but about:
  - A systematic quality management approach
  - Studying processes and identifying opportunities for defect elimination
  - Defect prevention approaches
  - Measuring output quality and improving it constantly

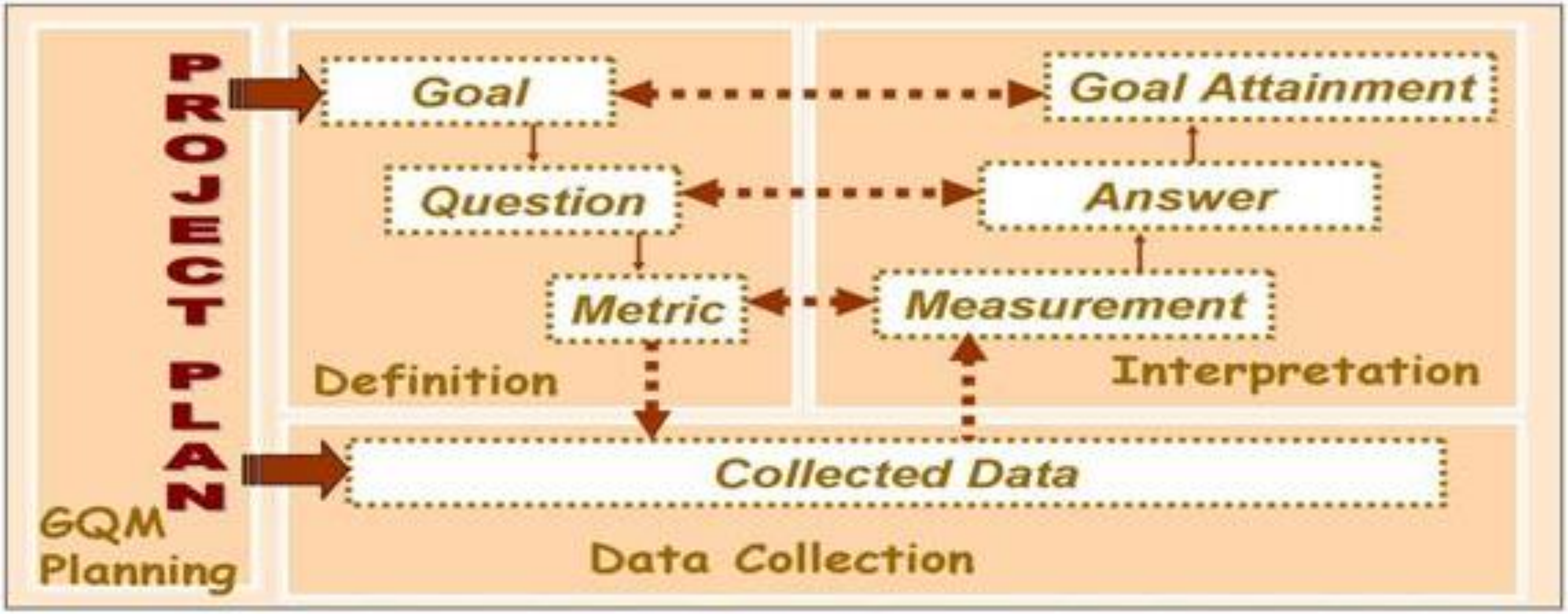
Comments on Process Variation

- Note that "reducing" process variation is a "factory view" of engineering development
  - Need to be careful about applying it to engineering processes
  - Each software product may vary, but be consistent in the engineering processes
- Most applicable for activities performed repeatedly, such as, writing code, running tests, creating releases, etc.
- Less applicable for activities that are different every time, such as, innovation, learning a domain, architecting a system
  - Many "creative" activities do have a repetitive component
  - Partly amenable to "systematic defect elimination" such as in design
- Simple criterion: Are there defects that can be eliminated by systematic process improvement?
  - Reducing variation eliminates some kinds of defects
  - Defect elimination is a two-outcome model—ignores excellence

GQM Approach for Defining and Using Metrics

The following is based on Goal-Question-Metric Software Acquisition Gold Practice at the DACS Gold Practices Web Site
(https://www.goldpractices.com/practices/gqm/)

Phases of GQM Implementation



Source: Solingen, "Experiences in Using the Goal/Question/Metric Paradigm" , 1998

Six Steps of GQM

- Steps 1-3: Definition
  - Use business goals to drive identification of the right metrics
- Steps 4-6: Data Collection and Interpretation
  - Gather the measurement data and make effective use of the measurement results to drive decision making and improvements

Six Steps of GQM
Steps 1-3: Definition

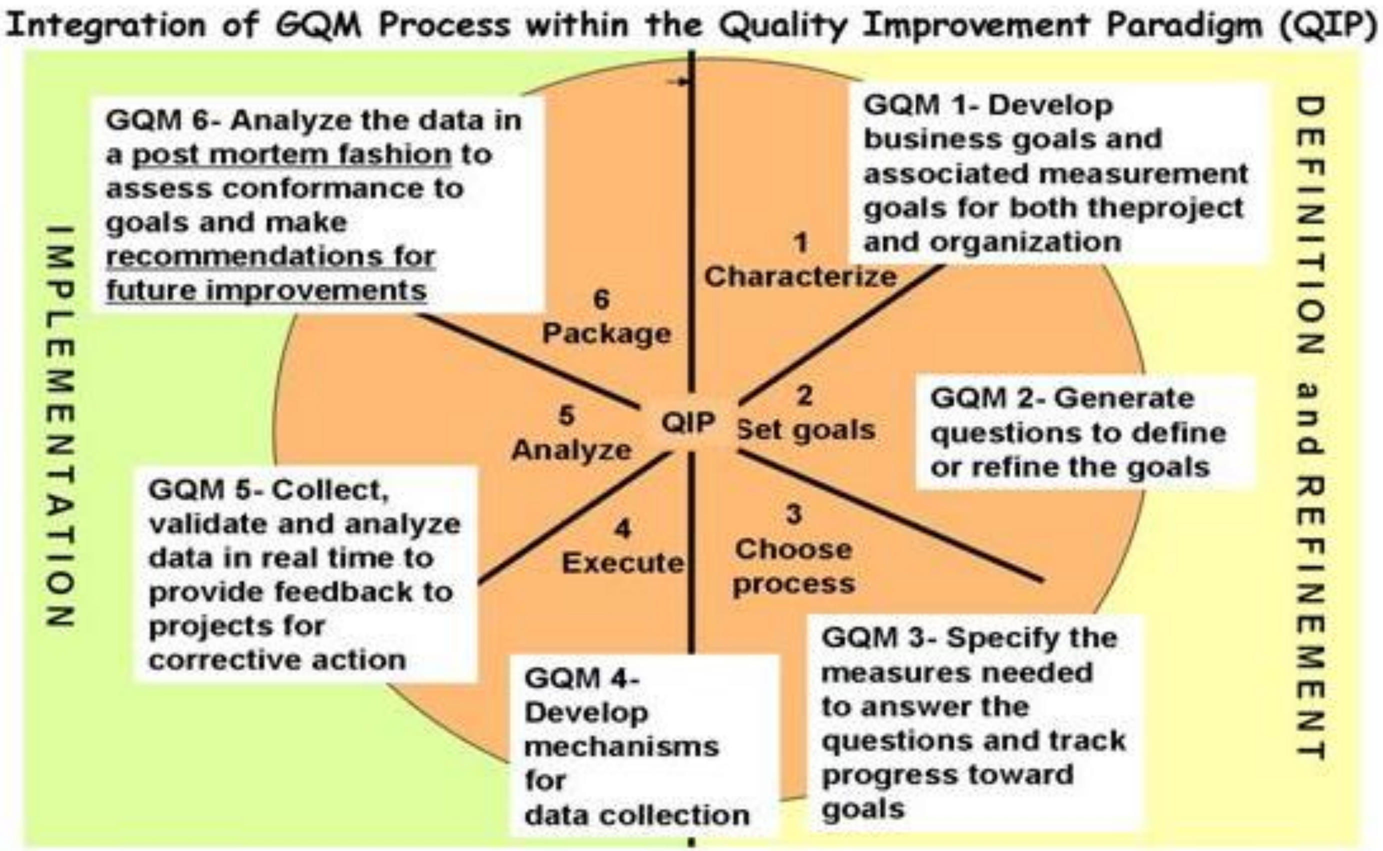
Use business goals to drive identification of the right metrics

1. **Develop** a set of corporate, division and project **business goals and associated measurement goals** for productivity and quality
2. **Generate questions** (based on models) that define those goals as completely as possible in a quantifiable way
3. **Specify** the **measures** needed to be collected to answer those questions and track process and product conformance to the goals

Six Steps of GQM
Steps 4-6: Data Collection and Interpretation

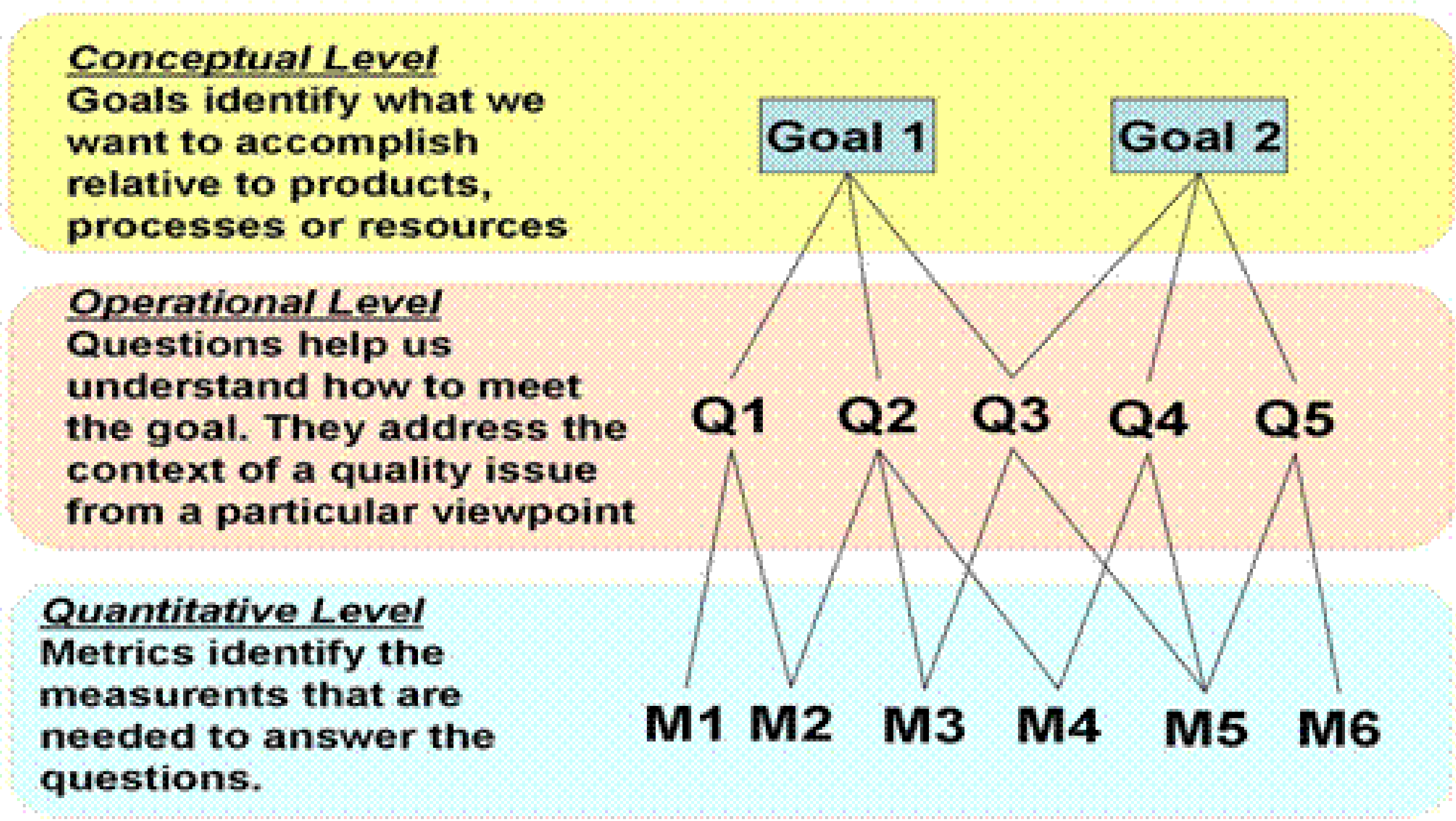
Gather the measurement data and make effective use of the measurement results to drive decision making and improvements

4. **Develop mechanisms** for data collection
5. **Collect, validate** and **analyze** the **data in real time** to provide feedback to projects for corrective action
6. **Analyze** the **data** in a **postmortem** fashion to assess conformance to the goals and to make recommendations for future improvements



Based on: Basili, "Using Measurement to Build Core Competencies in Software", DACS Course, 2005

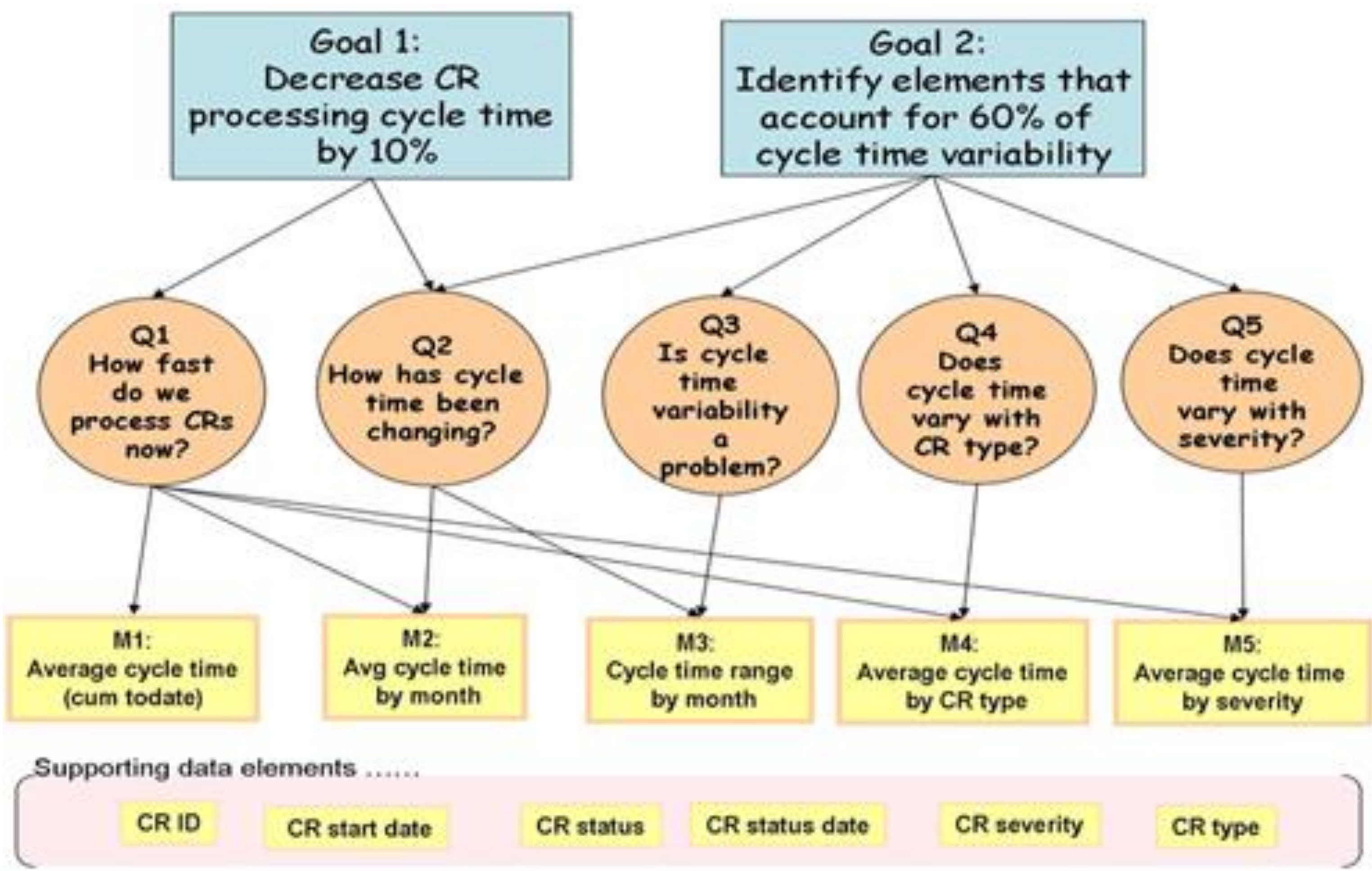
GQM Metrics Definition



Goals identify what we want to accomplish; ***questions***, when answered, tell us whether we are meeting the goals or help us understand how to interpret them; and the ***metrics*** identify the measurements that are needed to answer the questions and quantify the goal

(CR: Change Request)

Example



Defining Goals—PPE Template

- ***Purpose****:* Analyze some (objects: processes, products, other experience models) for the purpose of (why: characterization, evaluation, prediction, motivation, improvement)
- ***Perspective****:* with respect to (what aspect: cost, correctness, defect removal, changes, reliability, user friendliness, etc.) from the point of view of (who: user, customer, manager, developer, corporation, etc.)
- ***Environment****:* in the following context: (where: problem factors, people factors, resource factors, process factors, etc.)

IEEE-STD-1061-1998 Standard for Software Quality Metrics Methodology

Goal Example

- Analyze the (system testing method) for the purpose of (evaluation) with respect to a model of (defect removal effectiveness) from the point of view of the (developer) in the following context: the standard NASA/GSFC environment, i.e., process model [e.g., Software Engineering Laboratory (SEL) version of the waterfall model], application (ground support software for satellites), machine (running on a DEC 780 under VMS), etc.

IEEE-STD-1061-1998 Standard for Software Quality Metrics Methodology

Key Practices of GQM (p. 1 of 3)

- ***Get the right people involved in the GQM process***
- ***Set explicit measurement goals and state them explicitly***
- ***Don't create false measurement goals*** (for example, matching metrics you already have or are easy to get)
- ***Acquire implicit quality models from the people involved***

Key Practices of GQM (p. 2 of 3)

- ***Consider context***
- ***Derive appropriate metrics***
- ***Stay focused on goals when analyzing data***
- ***Let the data be interpreted by the people involved***
- ***Integrate the measurement activities with regular project activities***

Key Practices of GQM (p. 3 of 3)

- ***Do not use measurements for other purposes (such as to assess team member productivity)***
- ***Secure management commitment to support measurement results***
- ***Establish an infrastructure to support the measurement program***
- ***Ensure that measurement is viewed as a tool, not the end goal***
- ***Get training in GQM before going forward***

Conclusions

- Measurement starts with an operational definition of some quality attribute of interest
  - We need to put some effort into choosing appropriate measures and scales, and understanding their limitations
- Measurements have both systematic and random error
- Measurements must have both reliability and validity
  - Often, hard to achieve both
- A common error is confusing correlation with causation
- A major concern in process design is reducing process variation:
  - Six sigma is actually more about eliminating and identifying defects, and identifying opportunities for process improvement
  - Defects are NOT the sole concern in process design!
    - There are other quality attributes than defects and failures
  - Process optimization is oriented primarily towards repetitive activities
- GQM provides a method for identifying metrics from quality goals