2

Measurement Information Model

This chapter describes one of the fundamental measurement concepts of *Practical Software Measurement*, the Measurement Information Model. The Measurement Information Model provides a formal mechanism for linking defined information needs to software engineering processes and products that can be measured. Developing a measurement plan for a particular project requires an instantiation of the Measurement Information Model that is specific to the project's information needs. The model serves as a primary resource throughout the measurement process, as it guides the planning and implementation of data collection and analysis activities.

The Measurement Information Model establishes a defined structure for relating different measurement concepts and terms. As such, it establishes a foundation for the consistent definition of measurement terminology. Many terms used by the software measurement community lack widely agreed-on definitions. The term *metric* is probably the best example, as it is commonly misused to describe many different measurement concepts. Because defining and implementing a measurement program involves making many choices with respect to collecting, organizing, and analyzing data, success requires a systematic approach to considering and describing those choices. Without concise and consistent terminology, effective communication among measurement analysts, data providers, and measurement users is impossible. Because decision makers need to understand measurement information, consistent measurement terminology is mandatory. Some of the key terms are addressed below.

Figure 2-1 is an overview of how an information need evolves into a plan for generating the project's measurement products. Measurement planning begins with the recognition that a manager or engineer has a specific **information need** required to support project decision making. Project decisions are usually related to project planning and execution in an operational sense, but may also address strategic and organizational-level requirements. Data that helps to satisfy the defined information need can be obtained by measuring many different software process elements and product characteristics, called *software entities.* The **measurable concept** is an idea about the entities that should be measured in order to satisfy an information need. For example, a decision maker concerned with allocating budget and associated resources to a software task may believe that productivity is related to the type of software task that will be performed. Productivity is therefore the measurable concept that addresses the defined information need. Determining productivity requires that entities such as the software product and process be measured. There are many ways that productivity can be computed, so, at this stage productivity is still only a measurable concept. Eventually, the measurable concept will be formalized as a **measurement construct** that specifies exactly what will be measured and how the data will be combined to produce results that satisfy the information need. Two applicable measures in this example could be software size and effort. A **measurement procedure** defines the mechanics of collecting and organizing the data required to instantiate a measurement construct.

All of the applicable information needs, measurement constructs, and measurement procedures are combined into a **measurement plan**. Execution of the measurement plan produces the **information products** that respond to the project information needs. The information products are the collection of indicators, interpretations, and recommendations provided to the decision maker as an output of the measurement process.

A measurement plan can address a single information need. In most cases, however, it addresses multiple information needs. The plan defines which information needs are applicable to a particular project, how the software will be measured to satisfy those information needs, and how the measurement process will be resourced and managed.

As Figure 2-1 suggests, developing an effective measurement program requires a thorough understanding of the project's software information

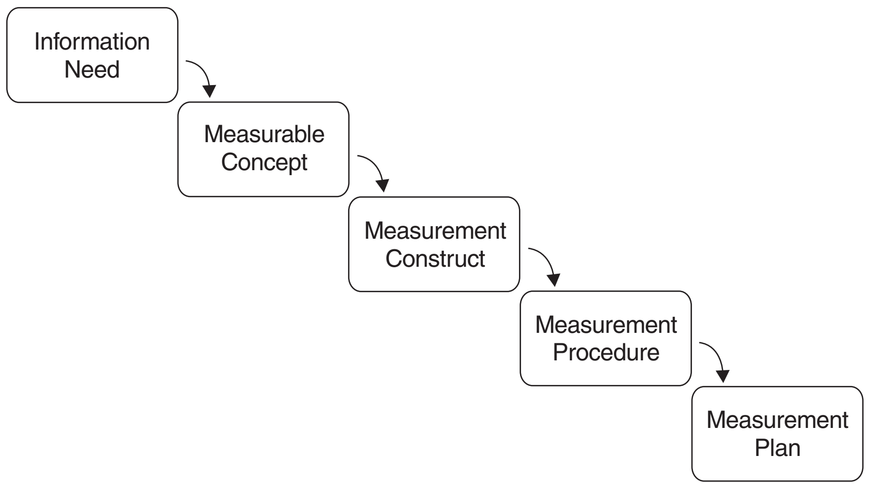

Figure 2-1 Evolution of an information need into a measurement plan

needs, a defined set of related measurable concepts, and knowledge of which software entities are available to be measured within the project.

The terms defined in PSM are derived from the Measurement Information Model documented in the international standard ISO/IEC 15939, “Software Measurement Process.” The ISO/IEC 15939 standard describes in a general sense how data is collected and organized to satisfy defined information needs. The reader’s knowledge of the software entities is presumed. However, the following sections explain the concepts of information needs and measurement constructs in more detail.

2.1 Information Needs

Information needs result from the efforts of engineers and managers to influence the outcomes of software engineering processes and activities. These technical and management processes may be defined and may operate at either the project or organizational levels. The desired outcomes of such processes usually are defined in terms of objectives. The information needs relate directly to both the established objectives and to the areas of concern that may impact the achievement of these objectives. These areas of concern are often referred to as **issues.** There are three types of issues:

- *Problems*—areas of concern that a project is currently experiencing or is relatively certain to experience
- *Risks*—areas of concern that could occur, but are not certain to occur
- *Lack of information*—areas of concern where the available information is inadequate to reliably predict project impact

Identifying something as an issue does not necessarily mean that it is a problem. In fact, thorough and continual identification and careful tracking of project risks minimize the potential for the emergence of serious problems that could negatively impact project success.

Most projects work toward achieving fixed objectives in terms of budget, schedule, quality, and functionality. Consequently, measurement at the project level tends to focus on providing information related to issues in these areas. While every project involves unique issues, experience shows that most software project information needs can be organized into common information categories. These categories directly relate to project issues that the project manager must manage on a day-to-day basis.

PSM is an issue-based approach to software measurement for project management. PSM defines seven common software information categories:

- *Schedule and Progress:* This information category addresses the achievement of project milestones and the completion of individual work units. A project that falls behind schedule can usually meet its delivery objectives only by eliminating functionality or sacrificing product quality.
- *Resources and Cost:* This information category relates to the balance between the work to be performed and personnel resources assigned to the project. A project that exceeds the budgeted effort usually can recover only by reducing software functionality or by sacrificing product quality.
- *Product Size and Stability:* This information category addresses the stability of the functionality or capability required of the software. It also relates to the volume of software delivered to provide the

required capability. Stability includes changes in functional scope or quantity. An increase in software size usually requires increasing the applied resources or extending the project schedule.

- *Product Quality*: This information category addresses the ability of the delivered software product to support the user's needs without failure. If a poor-quality product is delivered, the burden of making it work usually falls on the assigned maintenance organization.
- *Process Performance*: This information category relates to the capability of the supplier relative to project needs. A supplier with a poor software development process or low productivity may have difficulty meeting aggressive project schedule and cost objectives.
- *Technology Effectiveness*: This information category addresses the viability of the proposed technical approach. It addresses engineering approaches such as software reuse, use of commercial software components, reliance on advanced software development processes, and implementation of common software architectures. Cost increases and schedule delays may result if key elements of the proposed technical approach are not achieved.
- *Customer Satisfaction*: This information category addresses the degree to which products and services delivered by the project meet the customer's expectations. Indications of satisfaction may be obtained from customer feedback and the levels of customer support required.

Practical Software Measurement makes use of these common information categories to facilitate the identification and prioritization of a project's specific information needs. In practice, information related to similar information needs often can be addressed using similar measurable concepts, thus reducing the number of resources that must be applied to implement a viable measurement program.

2.2 Measurement Construct

A measurement construct is a detailed structure that links the things that can be measured to a specified information need. Figure 2-2 outlines the

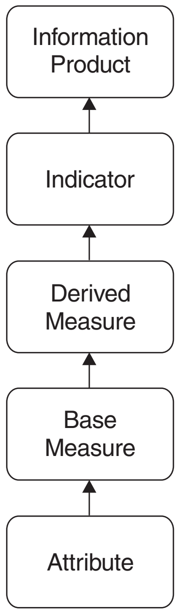

Figure 2-2 Levels of a measurement construct

basic structure of a measurement construct. The things that can actually be measured include specific attributes of software processes and products, such as size, effort, and number of defects. The measurement construct describes how the relevant software attributes are quantified and converted to indicators that provide a basis for decision making. A single measurement construct may involve three types, or levels, of measures: base measures, derived measures, and indicators. The measurement planner needs to specify the details of the measurement constructs to be used, as well as the procedures for data collection, analysis, and reporting, in the measurement plan. The better the design of the measurement construct and the better it relates the measured attributes of the software to the identified information need, the easier it is for the project manager to make informed, objective decisions.

At each of the three levels of measures—base measures, derived measures, and indicators—additional information content is added in the form of rules, models, and decision criteria. Figure 2-3 illustrates the structure of a measurement construct in more detail. Specific rules for assigning values,

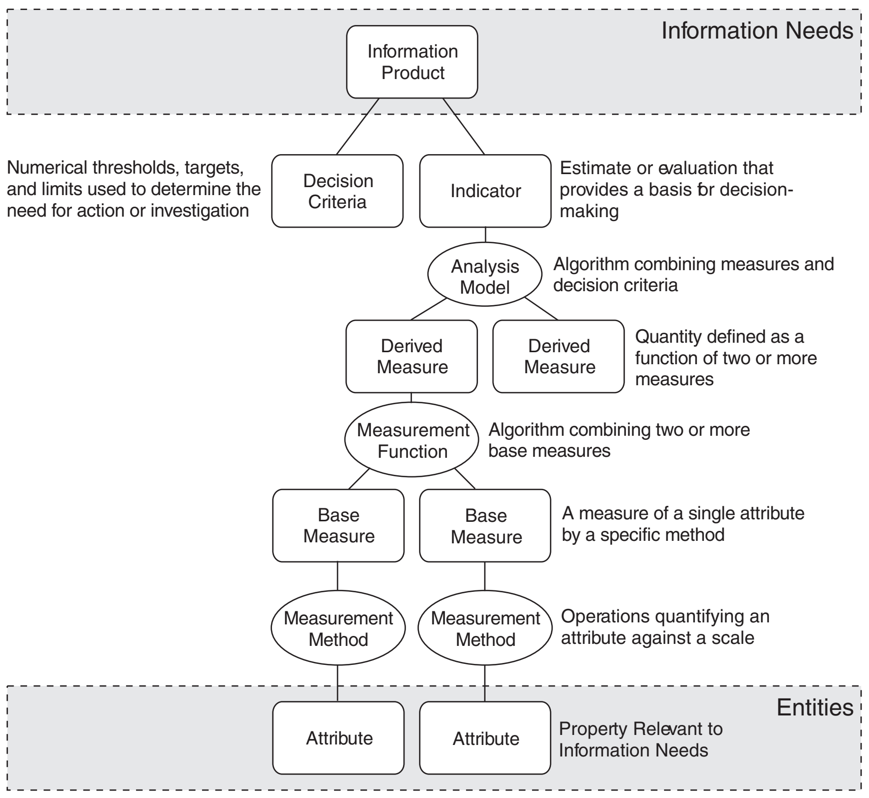
Information Needs
Information Product
Numerical thresholds, targets, and limits used to determine the need for action or investigation
Decision Criteria
Indicator
Estimate or evaluation that provides a basis for decision-making
Analysis Model
Algorithm combining measures and decision criteria
Derived Measure
Derived Measure
Quantity defined as a function of two or more measures
Measurement Function
Algorithm combining two or more base measures
Base Measure
Base Measure
A measure of a single attribute by a specific method
Measurement Method
Measurement Method
Operations quantifying an attribute against a scale
Entities
Attribute
Attribute
Property Relevant to Information Needs

Figure 2-3 Details of a measurement construct

and defining the measurement methods, measurement functions, and analysis models are associated with each level of measure. In formalizing a measurable concept as a measurement construct, the details of the actual processes and products of the software project must be considered, as well as the detailed description of the associated information need. The measurement construct must be able to be practically implemented given the project's characteristics, and it must satisfy the defined information need.

While the complexity implied in Figure 2-3 may seem intimidating, these details help ensure the implementation of an effective and efficient measurement program. The most important benefits of using well-defined measurement constructs include:

- Reducing redundancy by helping to identify a core set of base measures (or *data primitives*) that can serve many purposes
- Increasing accuracy and repeatability by ensuring that all essential aspects of the measurement approach are adequately defined
- Maximizing the value of the base measures by creating patterns of derived measures and indicators that can easily be recognized, reused, and adapted
- Documenting the link between the information need and how is it satisfied

Although Figure 2-3 is organized hierarchically, the process of defining a measurement construct is actually iterative. Chapter 3 describes the measurement planning process in more detail. Section 2.3 and Appendix A provide examples of measurement constructs that address specific information needs.

The individual components of the measurement construct as depicted in Figure 2-3 are described in the following sections.

Attribute (Measurable)

A measurable **attribute** is a distinguishable property or characteristic of a software **entity**. Entities include processes, products, projects, and resources. An entity may have many attributes, only some of which may be suitable to be measured. A measurable attribute is distinguishable either quantitatively or qualitatively by human or automated means. The first step in defining a specific instantiation of the Measurement Information Model is to select the software attributes most relevant to the measurement user's information needs. A given attribute may be incorporated in multiple measurement constructs supporting the same or different information needs.

Base Measure

A **base measure** is a measure of a single attribute defined by a specified measurement method. Executing the method produces a value for the measure. A base measure is functionally independent of all other measures and captures information about a single attribute. Data collection involves assigning values to base measures. Specifying the range and/or type of values that a base measure is expected to take on helps to verify the quality of the data collected. (Data verification is discussed in more detail in Chapter 4.) Each base measure is defined with the following characteristics:

Measurement Method

This is a logical sequence of operations, described generically, used in quantifying an attribute with respect to a specified scale. The operations may involve activities such as counting occurrences or observing the passage of time. The same measurement method may be applied to multiple attributes. However, each unique combination of an attribute and a method produces a different base measure. A measurement method may be implemented in two ways—either automated or manual. A **measurement procedure** describes the specific implementation of a measurement method within a given organizational context. Procedures may be documented either in the measurement plan or separately (see Chapter 3).

Type of Method The type of measurement method depends on the nature of the operations used to quantify a particular attribute. There are two types of methods:

- *Subjective*—quantification involving human judgment or rating. For example, relying on an expert to rate the complexity of functions as high, medium, or low is a subjective method of measurement.
- *Objective*—quantification based on numerical rules such as counting. These rules may be implemented by human or automated means. For example, lines of Ada code may be quantified by counting semicolons.

Objective measures typically provide more accuracy and repeatability than subjective methods. Where possible, objective methods of measurement are preferable

Scale

A scale is an ordered set of values, continuous or discrete, or a set of categories to which an attribute is mapped. The scale defines the range of possible values that can be produced by executing the measurement method.

The measurement method maps the magnitude of the measured attribute to a value on a scale. A unit of measurement often is associated with a scale.

Type of Scale The type of scale depends on the nature of the relationship between values on the scale. Four types of scales are commonly defined:

- *Ratio*—numeric data for which equal distances correspond to equal quantities of the attribute, where the value of 0 corresponds to none of the attribute. Counting lines of code produces a ratio scale with values ranging from 0 to (potentially) positive infinity.
- *Interval*—numeric data for which equal distances correspond to equal quantities of the attribute without the use of 0 values. Cyclomatic complexity (the number of logical paths through a software component) provides an interval scale because the minimum value possible is 1.
- *Ordinal*—discrete rankings. For example, a set of software units might be ordered in terms of the expected difficulty of implementing them: most difficult, second most difficult, and so forth.
- *Nominal*—categorical data. For example, a set of problem reports might be classified in terms of the origin of the problem: requirements, design, and so forth.

The type of measurement method usually affects the type of scale that can be used reliably with a given attribute. For example, subjective methods of measurement usually support only ordinal or nominal scales.

Unit of Measurement A unit of measurement is a particular quantity, defined and adopted by convention, with which other quantities of the same kind are compared in order to express their magnitude relative to that quantity. Only quantities expressed in the same units of measurement are directly comparable. Examples of units include the *hour* and the *meter*. Units of measurement often are not defined for measures determined by subjective methods or incorporating ordinal or nominal scales.

Derived Measure

A derived measure is a measure, or quantity, that is defined as a function of two or more base and/or derived measures. A derived measure captures information about more than one attribute. An example of a derived measure is a calculated value of productivity that is derived by dividing the base measure of hours of effort by the base measure of lines of code. Simple transformations of base measures (for example, taking the square root of a base measure) do not add information, thus do not produce derived measures. Normalization of data often involves converting base measures into derived measures that can be used to compare different entities. Derived measures are defined with the following characteristics:

Measurement Function

A measurement function is an algorithm or calculation performed to combine two or more values of base and/or derived measures. The scale and unit of the derived measure depend on the scales and units of the base measures from which it is composed, as well as how they are combined by the function. Division is the function used to produce the derived measure of productivity. The amount of product produced is divided by the amount of effort used to produce it.

Indicator

An indicator is a measure that provides an estimate or evaluation of specified attributes derived from an analysis model with respect to defined information needs. Indicators are the basis for measurement analysis and decision making. Indicators are, therefore, what should be presented to measurement users. Measurement is always based on imperfect information, so quantifying the uncertainty, accuracy, or importance of indicators is an essential component of presenting the actual indicator value. Indicators often are given the name of the measurable concept they implement. Indicators are defined with the following characteristics:

Analysis Model

This is an algorithm or calculation involving two or more base and/or derived measures with associated decision criteria. An analysis model is based on an understanding of, or assumptions about, the expected relationship between the component measures and their behavior over time. Analysis models produce estimates or evaluations relevant to defined information needs. The scale and method of measurement affect the choice of analysis techniques or models used to produce indicators. As an extreme example, computing the average of categorical data does not make sense.

Decision Criteria

These are numerical thresholds, targets, and limits used to determine the need for action or further investigation or to describe the level of confidence in a given result. Decision criteria help to interpret the measurement results. Decision criteria may be based on a conceptual understanding of expected behavior or calculated from data. Decision criteria may be derived from historical data, plans, and heuristics or computed as statistical control limits or statistical confidence limits.

Other Terminology

A few other issues related to terminology should be addressed before leaving the discussion of a measurement construct. The term **measures** may be used collectively to refer to base measures, derived measures, and/or indicators. The term **data** may be used collectively to refer to the values assigned to base measures and derived measures. Data collection refers to the assignment of values to base measures. In contrast, values of derived measures and indicators are computed from base and/or derived measures. Corresponding measurable concepts, measurement constructs, and indicators often are given the same name. Care must be used to distinguish between them as they are applied.

The measurement information model suggests two general principles for dealing with the question of how much measurement should be standardized within a project or organization. These principles are as follows:

- Standardize on base measures rather than indicators. The same set of base measures can be combined in many ways to produce different indicators that address different information needs. Base measures are tied to specific entities—which tend to be relatively persistent.
- Promote flexibility in the indicators. No single view of the data can effectively serve all levels and types of potential decision makers at all times throughout the project's life cycle. Indicators are tied to information needs—which tend to change frequently.

Measurement constructs are central to the measurement process. During the Plan Measurement activity, the necessary measurement constructs are designed, and specific procedures for collecting and analyzing the data are defined. During the Perform Measurement activity, these procedures are executed to collect data, assign values to the measurement constructs, and develop the information product. Careful attention to the Measurement Information Model throughout these steps helps obtain the maximum benefit from a project's or organization's measurement investment.

The key components of a measurement construct include measurement methods, base measures, measurement functions, derived measures, analysis models, and indicators. Figure 2-4 depicts the relationship between these components mathematically.

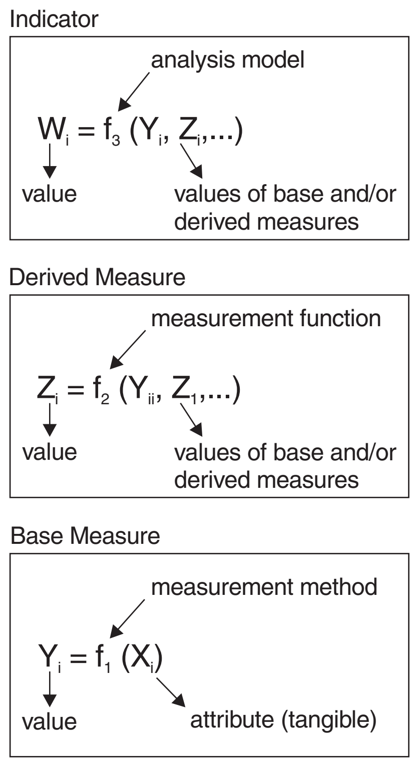

Figure 2-4 Mathematical depiction of a measurement construct

mcgarry02.qxd 9/7/01 1:27 PM Page 26

2.3 Measurement Construct Examples

For any given measurable concept, many different measurement constructs can be devised. Several examples of measurement constructs that address common project information needs are provided in the following sections. Bear in mind that these are generic examples. Moreover, these examples illustrate the variety of options supported by the Measurement Information Model and are not necessarily examples of universally recommended measurement constructs.

Productivity Example

The decision maker in the example depicted in Figure 2-5 needs to identify an expected software productivity level as the basis for project planning. The measurable concepts are that productivity from past projects can be used to validly estimate future projects and that productivity is related to the

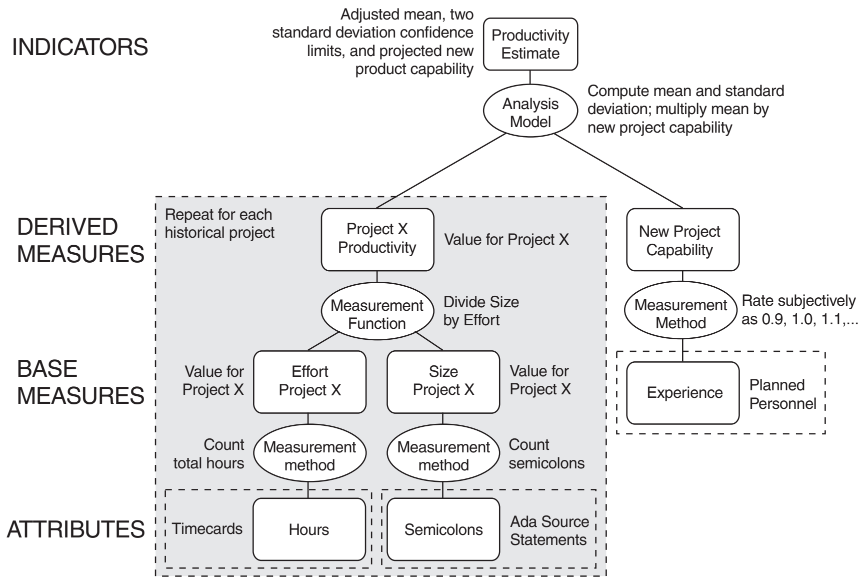

Figure 2-5 Measurement construct—Productivity example

effort expended and amount of software produced. Thus, effort and code are the measurable software entities of concern. Specific attributes of those entities must be selected for quantification and a function designated for combining them into a derived measure for each project in the database.

Regardless of how the productivity number is determined, the uncertainty inherent in software engineering means that there is a high probability the estimated productivity will not be exactly what is achieved. Estimating productivity based on historical data, however, enables the computation of confidence limits that help to assess how close actual results are likely to come to the estimated value.

Quality Example

The decision maker in the example depicted in Figure 2-6 needs to evaluate detailed design quality during the implementation of the software design activity. The measurable concept is that design quality is related to

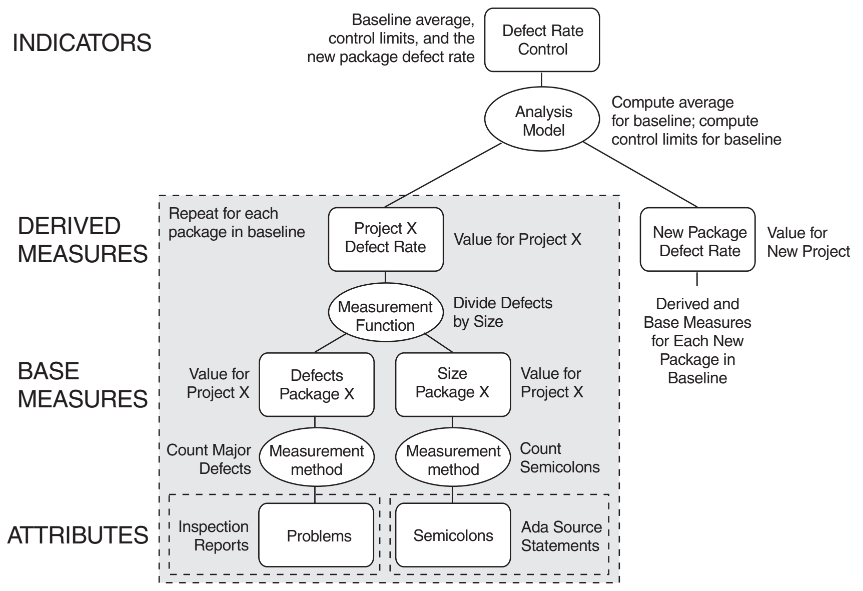

Figure 2-6 Measurement construct—Quality example

the amount of design produced and the number of defects found. Thus, the design packages and the lists of defects are the entities of concern. Computing the defect rate can normalize the quality of the design packages allowing a basic comparison. Thus, data for the base measures must be collected and the derived measure computed for each package as it is reviewed.

Since the defect rate is expected to be somewhat different for every package, control limits can be computed from a baseline set of packages to determine if the defect rate on a new package is different enough from the average to warrant concern.

Coding Progress Example

The decision maker in the example depicted in Figure 2-7 needs to evaluate whether or not the rate of code generation on a project is sufficient to meet projected schedule objectives. The measurable concept is that

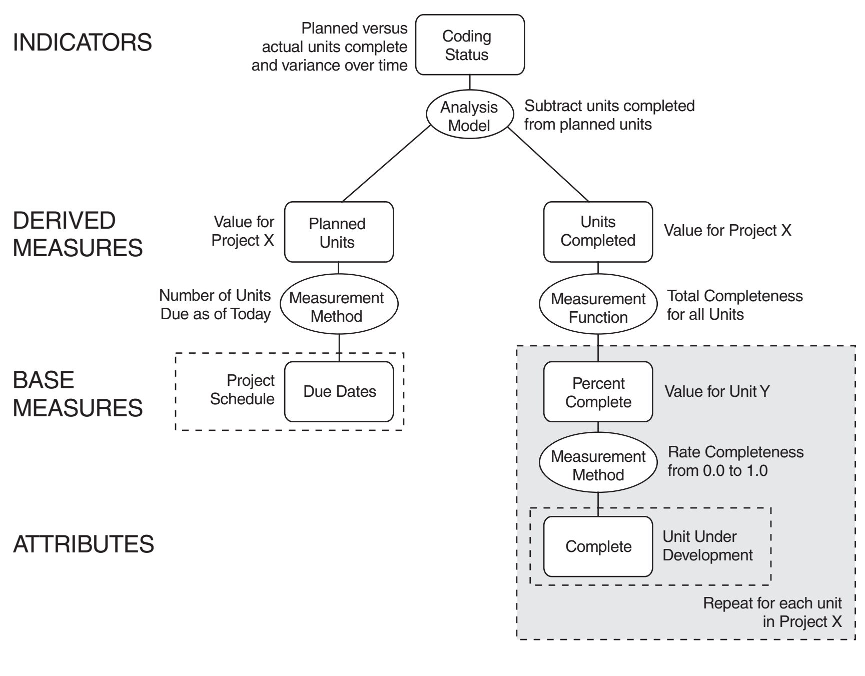

Figure 2-7 Measurement construct—Coding progress example

progress is related to the amount of work planned and the amount of work completed. Planned work items and completed work items are the entities of concern. This example assumes that the degree of completion of each software unit is reported by the developer assigned to it. Thus, data for the base measures must be collected, and the derived measure computed for each work item in the plan. A simple numerical threshold is used as a decision criterion rather than statistical limits.

Since the status of units is a subjective assessment, the indicator is subject to the influence of wishful thinking on the part of the programmers. A more reliable approach would be to count only units completed as defined by explicit exit criteria.