

Back to the Basics: Measurement and Metrics

Tim Perkins and Roald Peterson
Software Technology Support Center/SAIC

Larry Smith
Software Technology Support Center

Measurements and metrics are key tools to understanding the behaviors, successes, and failures of our programs and projects. This article highlights the basic principles of measures and metrics and encourages the reader to improve his or her use of these tools. The article is adapted from [1].

According to Tom DeMarco, “You cannot control what you cannot measure” [2]. Imagine going on a road trip of over a thousand miles. This is easy because most of us really have done this several times. Now imagine that your car has no speedometer, no odometer, no fuel gauge, and no temperature indicator. Imagine also that someone has removed the mile markers and road signs from all the roads between you and your destination. Just to complete the experiment, remove your watch.

What was once a simple journey becomes an endless series of guesses, fraught with risks. How do you know where you are, how far you have gone, or how far you have to go? When do you gas the car? Should you stop here or try to make the next town before nightfall? You could break down, run out of gas, be stranded, take the wrong road, bypass your destination, or waste time trying to find your location and how to reach your destination. Clearly, some method of measuring certain indicators of progress is essential for achieving a goal.

Imagine again going on a road trip. This time the cockpit of the car is filled with instruments. In addition to what you have been accustomed to in the past, there are now the following gauges:

- Speed in feet and yards per second, and as a percentage of c (light speed).
- Oil pressure in millibars.
- Estimated time to deplete or recharge the battery.
- Fuel burn rate and fuel weight.
- Oil viscosity and transparency indicators.
- Antifreeze temperature and pressure.
- Engine efficiency.
- Air conditioning system parameters (pressures, temperatures, efficiency).
- Elevation, rate of climb, heading, accelerometers for all directions.
- Indicators for distance and time to destination and from origin.
- Inside air temperatures for eight different locations in the car.

Also, there are instruments to count how many cars pass, vibration levels, and

sound pressure levels within and outside the car. There are weather indicators for outside temperature, humidity, visibility, cloud ceiling, ambient light level, true and relative wind speeds and directions, warning indicators for approaching storms and seismic activity, etc. Along the roads will be markers for every hundredth mile and signs announcing exits every quarter mile for five miles before an exit is reached. Signs in five-mile-per-hour increments will announce speed changes.

“Metrics are measurements of different aspects of an endeavor that help us determine whether we are progressing toward the goal of that endeavor.”

To some, this may seem like a dream come true, at least the cockpit part. However, careful consideration will soon reveal that the driver will be inundated and quickly overwhelmed with unnecessary, confusing data. Measurement, in itself, is no prescription for achieving a goal. It can even make the goal unattainable.

Introduction

Metrics are measurements of different aspects of an endeavor that help us determine whether we are progressing toward the goal of that endeavor. They are used extensively as management tools to provide some calculated, observable basis for making decisions. Some common metrics for projects include schedule deviation, remaining budget and expenditure rate, presence or absence of specific types of problems, and milestones achieved. Without some way to accurately track

budget, time, and work progress, a project manager can only make decisions in the dark. Without a way to track errors and development progress, software development managers cannot make meaningful improvements in their processes. The more inadequate our metrics program, the closer we are to herding black cats in a dark room. The right metrics, used in the right way, are absolutely essential for project success.

Too many metrics are used simply because they have been used for years, and people believe they might be useful [3]. Each metric should have a purpose, providing support to a specific decision-making process. Leadership too often dictates metrics. A team under the direction of leadership should develop them. Metrics should be used not only by leadership but also by all the various parts of an organization or development team. Obviously, not all metrics that are useful to managers are useful to the accounting people or to developers. Metrics must be tailored to the users. The use of metrics should be defined by a program describing what metrics are needed, by whom, and how they are to be measured and calculated. The level of success or failure of your project will depend in large part on your use or misuse of metrics – on how you plan, implement, and evaluate an overall metrics program.

While this article introduces and describes key metrics ideas and processes and can point you in the right direction, it is recommended that you gain more insight, depth, and specific examples by downloading and reading the material listed in “Additional Reading” in the online version of this article at <www.stsc.hill.af.mil/crosstalk>. Of particular value to Department of Defense users is [4] better known as the “Practical Software and System Measurement Guidebook,” where a more specific and detailed terminology and methodology can be found.

Process Description

Metrics are not defined and used solely,

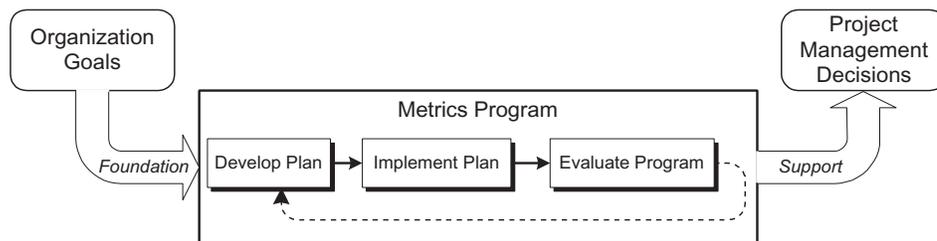


Figure 1: Metrics Program Cycle



Figure 2: Basili's Goal, Question, Metric Paradigm

but are part of an overall metrics program. This program should be based on the organization's goals and should be carefully planned, implemented, and regularly evaluated for effectiveness. The metrics program is used as a decision support tool.

In relation to project management metrics, if the information provided through a particular metric is not needed for determining status or direction of the project, it is probably not needed at all. Process-related metrics, however, should not necessarily be dismissed so harshly since they indicate data useful in improving performance across repeated applications. The role of the metrics program in the organization and its three major activities are shown in Figure 1.

Developing a Metrics Program Plan

The first activity in developing a metrics program is planning. Metrics planning is usually based on the goal-question-metric (GQM) paradigm developed by Victor Basili (see Figure 2). The GQM paradigm is based on the following key concepts [3]:

1. Processes, including software development, program management, etc., have associated goals.
2. Each goal leads to one or more questions regarding the accomplishment of the goal.
3. Each question leads to one or more metrics needed to answer the question.

4. Each metric requires two or more measurements to produce the metric (e.g., miles per hour, budget spent vs. budget planned, temperature vs. operating limits, actual vs. predicted execution time, etc.).
5. Measurements are selected to provide data that will accurately produce the metric.

The planning process is comprised of the three sub-activities implementing the GQM paradigm and one that defines the data collection process. Each of these is discussed in the following sections.

Table 1 shows two examples of goals and their related questions and metrics. Note that there could be one or more metrics associated with each question. As the initial list of questions and metrics is written and discussed, the goal is usually refined, which then causes a further refinement in the accompanying questions and metrics.

Define Goals

Planning begins with well defined, validated goals. Goals should be chosen and worded in such a way that they are verifiable; that is, their accomplishment can be measured or observed in some way. Goals such as meeting a specific delivery schedule are easily observable. Requirements stating "software shall be of high quality" are highly subjective and need further definition before they can be used

as valid goals.

You may have to refine or even derive your own goals from loosely written project objectives. The selection or acceptance of project goals will determine how you manage your project, and where you put your emphasis. Goals should meet the following criteria:

- They should support the successful accomplishment of the project's overall or system-level goals.
- They should be verifiable, or measurable in some way.
- They should be defined in enough detail to be unambiguous.

Derive Questions

Each goal should evoke questions about how its accomplishment can be measured. For example, completing a project within a certain budget may evoke questions such as these: What is my total budget? How much of my budget is left? What is my current spending rate? Am I within the limits of my spending plan?

Goals related to software time, size, quality, or reliability constraints would evoke different questions. It should be remembered that different levels and groups within the organization might require different information to measure the progress in which they are interested. Questions should be carefully selected and refined to support the previously defined project goals. Questions should exhibit the following traits:

- Questions only elicit information that indicates progress toward or completion of a specific goal.
- Questions can be answered by providing specific information. (They are unambiguous.)
- Questions ask all the information needed to determine progress or completion of the goal.

Once questions have been derived that elicit only the complete set of information needed to determine progress, metrics must be developed that will provide that information.

Develop Metrics

Metrics are the information needed to answer the derived questions. Each question can be answered by one or more metrics. These metrics are defined and associated with their appropriate questions and goals. Each metric requires two or more measurements. Measurements are those data that must be collected and analyzed to produce the metric.

Measurements are selected that will provide the necessary information with the least impact to the project workflow.

Table 1: Goals and Their Related Questions and Metrics

	Common Example	Product Example
Goal	Run competitively in a 10 kilometer (10K) race.	Improve customer satisfaction with the current release of the product.
Questions and Metrics	<p>What is a competitive time for an individual of my age and rank?</p> <ul style="list-style-type: none"> • Age and ranking figures for run times. <p>Am I capable of running at a competitive time?</p> <ul style="list-style-type: none"> • Time to complete 10K, post-run recovery time, etc., repeated over each practice run. <p>What current injuries are impacting my ability to race?</p> <ul style="list-style-type: none"> • Injury prognosis and recovery time. <p>Am I sustaining my health and weight by eating and sleeping properly?</p> <ul style="list-style-type: none"> • Hours of sleep per night, weight, dietary intake, etc. 	<p>Are customers buying our product?</p> <ul style="list-style-type: none"> • Sales rate (up or down) as compared with competing products, product return rate, etc. <p>What are the key attributes of customer satisfaction?</p> <ul style="list-style-type: none"> • Metrics related to reliability, safety, functionality, performance, etc. <p>How satisfied are our customers (in relation to the above attributes)?</p> <ul style="list-style-type: none"> • Customer survey data, defects reported, etc. <p>How are we resolving problems that affect customer satisfaction?</p> <ul style="list-style-type: none"> • Defect resolution rate, post-release defect density, etc.

Figure 3 summarizes the process of turning measurements into goal status.

Choosing measures is a critical and nontrivial step. Measurements that require too much effort or time can be counterproductive and should be avoided. Remember, just because something can be measured does not mean it should be. An in-depth introduction to measurements, “Goal-Driven Software Measurement – A Guidebook,” has been published by the Software Engineering Institute and is available as a free download [5].

In addition to choosing what type of data to collect or measure, the methods of processing or analysis must also be defined in this step. How do you turn the measurements into a meaningful metric? How does the metric then answer the question? The analysis method should be carefully documented. Do not assume that it is obvious.

This activity is complete when you know exactly what type of data you are going to collect (what you are going to measure and in what units), how you are going to turn that data into metrics (analysis methods), and in what form (units, charts, colors, etc.) the metrics will be delivered.

Define the Collection Process

The final step of the metrics planning process is to determine how the metrics will be collected. At a minimum, this part of the plan should include the following:

- What data is to be collected?
- What will be the source of the data?
- How is it to be measured?
- Who will perform the measurement?
- How frequently should the data be collected?
- Who will the derived metrics be delivered to, and in what format?

Implementing a Metrics Program

A good rule of thumb to follow when starting a measurement program is to keep the number of measurements between five and 10. If the metrics program is well planned, implementing the program should be reduced to simply following the plan. There are four activities in the metrics implementation cycle, shown in Figure 4 [6].

Data is collected at specific intervals according to the plan. Data is then validated by examining it to ensure it is the result of accurate measurements, and that the data collection is consistent among members of the group if more than one individual is collecting it. In other words, is it being measured in the

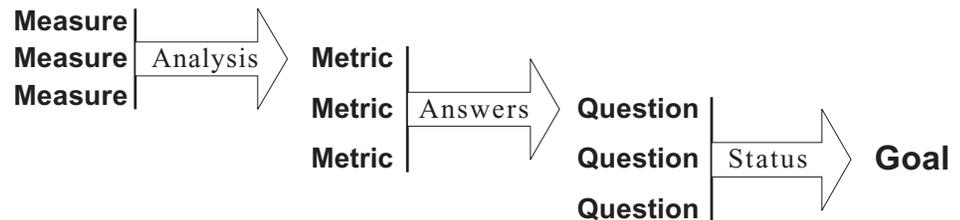


Figure 3: *Goal, Question, Metrics Examples*

same way, at the same time, etc.? Once the data is determined to be valid, the metrics are derived by analyzing the data as documented in the metrics program plan. Metrics are then delivered to appropriate individuals and groups for evaluation and decision-making activities. This process is repeated until the project is complete.

Evaluating a Metrics Program

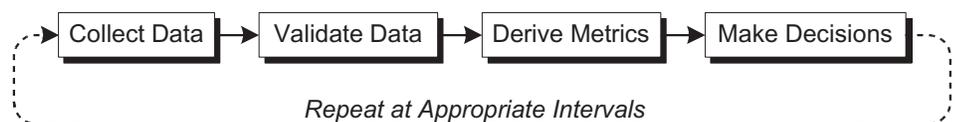
It is likely that a metrics program will not be perfect in its first iteration. Soon after its initial implementation and at regular

“Too many metrics are used simply because they have been used for years, and people believe they might be useful.”

intervals after that, the metrics program should be evaluated to determine if it is meeting the needs of the metrics users, and if its implementation is flowing smoothly. If metrics prove to be insufficient or superfluous, the program plan should be modified to provide the necessary information and remove any unneeded activity. The objective of a metrics program is to provide sufficient information to support project success while keeping the metrics program as simple and unobtrusive as possible. The following are areas that should be considered when reviewing a metrics program:

- Adequacy of current metrics.
- Superfluity of any metrics or measures.
- Interference of measurements with project work.
- Accuracy of analysis results.
- Data collection intervals.

Figure 4: *Metrics Implementation Cycle*



- Simplification of the metrics program.
- Changes in project or organization goals.

Metrics Repository

A final consideration is establishing a metrics repository where metrics history is kept for future projects. The availability of past metrics data can be a gold mine of information for calibration, planning estimates, benchmarking, process improvement, calculating return on investment, etc. At a minimum, the repository should store the following:

- Description of projects and their objectives.
- Metrics used.
- Reasons for using the various metrics.
- Actual metrics collected over the life of each project.
- Data indicating the effectiveness of the metrics used.

Measurement and Metrics Checklist

This checklist is provided to assist you in developing a metrics program, and in defining and using metrics. If you cannot answer a question affirmatively, you should carefully examine the situation and take appropriate action. The checklist items are divided into three areas: developing, implementing, and reviewing a metrics program.

Developing a Metrics Program

- Is your use of metrics based on a documented metrics program plan?
- Are you using the GQM paradigm in developing your metrics?
- Are your metrics based on measurable or verifiable project goals?
- Do your goals support the overall system-level goals?
- Are your goals well defined and unambiguous?
- Does each question elicit only information that indicates progress toward or completion of a specific goal?

- Can questions be answered by providing specific information? (Is it unambiguous?)
- Do the questions ask for all the information needed to determine progress or completion of the goal?
- Is each metric required for specific decision-making activities?
- Is each metric derived from two or more measurements (e.g., remaining budget vs. schedule)?
- Have you documented the analysis methods used to calculate the metrics?
- Have you defined those measures needed to provide the metrics?
- Have you defined the collection process (i.e., what, how, who, when, how often, etc.)?

Metrics Program Implementation

- Does your implementation follow the metrics program plan?
- Is data collected the same way each time it is collected?
- Are documented analysis methods followed when calculating metrics?
- Are metrics delivered in a timely man-

ner to those who need them?

- Are metrics being used in the decision-making process?

Metrics Program Evaluation

- Are the metrics sufficient?
- Are all metrics or measures required, that is, non-superfluous?
- Are measurements allowing project work to continue without interference?
- Does the analysis produce accurate results?
- Is the data collection interval appropriate?
- Is the metrics program as simple as it can be while remaining adequate?
- Has the metrics program been modified to adequately accommodate any project or organizational goal changes?◆

References

1. Software Technology Support Center. Condensed Version (4.0) of Guidelines for Successful Acquisition and Management of Software-Intensive Systems. Hill Air Force Base, Utah: Software Technology Support Center,

Feb. 2003.

2. DeMarco, Tom. Controlling Software Projects. New York: Yourden Press, 1982.
3. Perkins, Timothy K. "The Nine-Step Metrics Program." CROSSTALK, Feb. 2001 <www.stsc.hill.af.mil/crosstalk/2001/feb/perkins.asp>.
4. Bailey, Elizabeth, et al. Practical Software Measurement: A Foundation for Objective Project Management Ver. 4.0b1. Severna Park, MD: Practical Software and Systems Measurement, Mar. 2003 <www.psmc.com/PSMGuide.asp> under "Products."
5. Park, Robert E., et al. Goal-Driven Software Measurement – A Guidebook. Pittsburgh, PA: Software Engineering Institute, Aug. 1996 <www.sei.cmu.edu/publications/documents/96.reports/96.hb.002html>.
6. Augustine, Thomas, et al. "An Effective Metrics Process Model." CROSSTALK June 1999 <www.stsc.hill.af.mil/crosstalk/1999/jun/augustine.asp>.

About the Authors



Tim Perkins has been involved in software process improvement for the past 11 years, including leading the effort to initiate software process improvement at the then five Air Force Air Logistics Centers. As the Software Engineering Process Group leader at the Software Engineering Division at Hill Air Force Base, Utah, he led the division in reaching Capability Maturity Model® (CMM®) Level 3. The division has gone on to achieve CMM Level 5. Perkins is Acquisition Professional Development Program Level 3 certified in Project Management and System Planning, Research, Development, and Engineering.

**Software Technology Support Center
OO-ALC/MASE
6022 Fir Ave.
Bldg. 1238
Hill AFB, UT 84056
Phone: (801) 775-5736
Fax: (801) 777-8069
E-mail: tim.perkins@hill.af.mil**



Roald E. Peterson is a senior systems engineer with Science Applications International Corporation. He has 22 years of electronic systems development experience, specializing in communications, architecture, and software development. Peterson was an editor and contributor for the "Guidelines for the Successful Acquisition and Management (GSAM) of Software Intensive Systems" and is the author of the "Condensed GSAM Handbook." He has a bachelor's degree in physics and master's degrees in computer resources management and electrical engineering.

**Science Applications
International Corporation
920 W. Heritage Park Blvd.
Suite 210
Layton, UT 84041
Phone: (801) 774-4705
Fax: (801) 728-0300
E-mail: roald.e.peterson@saic.com**



Larry Smith is a senior software engineer and project manager for the Air Force's Software Technology Support Center at Hill Air Force Base, Utah. He provides software engineering, software process improvement, and project management consulting for the U. S. Air Force and other Department of Defense organizations as well as commercial and nonprofit organizations. Smith is a faculty member at the University of Phoenix. He is also certified by the Project Management Institute as a Project Management Professional. Smith has a bachelor's degree in electrical engineering and a master's degree in computer science.

**Software Technology Support Center
OO-ALC/MASE
6022 Fir Ave.
Bldg. 1238
Hill AFB, UT 84056
Phone: (801) 777-9712
Fax: (801) 777-8069
E-mail: larry.smith4@hill.af.mil**