Making Software Measurement *Really* Work: #1 Aligning Measurement Expectations

The first in a Series of Measurement Articles

By Carol A. Dekkers, CFPS, CMC
Quality Plus Technologies, Inc.
Email: dekkers@qualityplustech.com

Published in April 2000 in
IT Metric Strategies by
Cutter Corporation, USA.

Introduction

When it comes to measurement, the IT industry acts strangely. While other industries depend on measurement, tracking, and control as keys to profitability, the IT industry has yet to embrace measurement on a widespread basis. Even when it recognizes the merits of software measurement, the expectations for it are often unrealistic. Software practitioners want a silver-bullet metric that can answer any development question and do it to several-decimal-point accuracy. Predictably, software measurement doesn't match these expectations and, thus, is usually abandoned before it can deliver a return on investment. This doesn't have to be the case – software measurement can deliver value even when the measures are subjective (as with customer satisfaction ratings) or when the measures are imperfect (as with defect tracking).

This article addresses the mistaken notion of measurement or a particular metric being a silver bullet – a notion that left untapped can impede your organization from ever getting started with measurement. Future articles will focus on other aspects of measurement and progress from setup through implementation and growth of a measurement program.

... No Silver Bullets

Those of us with statistical or engineering backgrounds will always attempt to make measurement into an exact science (recall college labs where data outliers on research graphs were too difficult to explain and therefore were "erased" ?) In the real world of Information Technology, however, measurement doesn't always translate into predicable outcomes and not everything that can be measured should necessarily be.

Measurement consists of taking a series of observations about a process or product and analyzing the data to indicate where positive changes might be made. It is important to realize that just because something can be measured to the nth degree of accuracy does not make it valuable to measure – there needs to be a purpose and a method behind the measure before it will be useful.

The first step to creating a successful measurement program is to realign your and your company's expectations about software measurement:

1. **Follow the Goal-Question-Metric (GQM)** approach to software measurement introduced by Victor Basili of the University of Maryland. This approach forces companies to clearly identify their **strategic goals** for metrics and to **pose questions** that will track whether or not the goals are being met. Only then are the **metrics needed** to answer the questions identified and data collection mechanisms put into place. The resulting metrics necessarily depend on the specific goals and questions of the organization. Within the SEI Capability Maturity Model for software are a number of Level-3 key process areas that can form the basis of an organization's goals/questions/metrics. *(Note: A new book featuring a foreword by Victor Basili was recently published by McGrawHill: "The Goal/Question/Metric Method" by Rini van Solingen and Egon Berghout.).*

   This is an area that is often glossed over in an organization's rush to establish a solid metrics program quickly. DON'T skip the proper planning – this is critical. In the same way that skipping software requirements leads to products that do not meet customer needs, skipping measurement program requirements (Goal / Question / Metric) will lead to a measurement program that does not meet its customer needs. There is a great deal more to say about this topic than can be provided here, suffice to say that GQM is not a scientific approach, but a rational planned approach that will lead to higher success rates for measurement programs.

2. **Communicate early and often that there is no "silver-bullet" software metric, just as there is no silver-bullet accounting metric.** Defects, functional size, project duration, and work effort all measure a different aspect of software development, and they are not interchangeable. No single measure or single combination metric will satisfy all goals or answer all measurement questions — one must choose the metric suitable for each specific question. Once the specific, measurable goals, questions and metrics have been identified, select the most appropriate metric designed for the purpose. In the same way that a toolbox contains many tools, each specifically designed to serve a particular use, a measurement toolbox should contain specific measures selected to suit your specific needs. There is no Swiss army knife of metrics – you need to select the measure that best fits your needs be it defects, function points,

number of objects, lines of code, customer satisfaction, etc. – each is intended to measure a different aspect of software development.

3. **Learn about the available metrics and what they mean before implementing them in an organization.** For example, work effort is a function of many variables, including software size, implementation technology, development tools, skills, hardware platforms, degree of reuse, tasks to be done, and many others. As such, no single variable can accurately predict work effort; yet there is often an expectation that a single variable (for example, degree of reuse) can accurately predict effort. If one of your Goals is to increase estimating capability, it is also wise to research the available automated tools on the market and talk to actual users (not just tool vendors) about how their chosen tools works within their particular environment. Note that not all estimating tools address the same problem – some provide probabilistic estimates of work effort and cost, while others provide hourly breakdowns of predicted work effort. Which one will best suit your needs? It depends on your goals and what questions you need to answer.

4. **Plan a measurement program by using metrics and measures in the manner for which they are intended, and ensure that there is a common understanding of the chosen measures.** For example, functional size reflects the size of the software based on its functional user requirements, not the physical size of software. (Physical size of software is often expressed in lines of code.) Together with other variables, it can be used as a technology-independent measure of software size in order to predict effort or cost in software estimation models. However, functional size is not the right measure for predicting data access storage device needs — these depend on the technology and physical space taken up by the software and the volume of data and are better measured with other units. There is an abundance of information on the internet about various software metrics from organizations such as the Quality Assurance Institute (www.qaiusa.com), American Society for Quality (www.asq.org), and the International Function Point Users Group (www.ifpug.org).

5. **Remember that the accuracy of a metric is a function of the least accurate component measure it involves.** People often run into measurement difficulty when they assign several decimal places of accuracy to metrics that are derived from a series of relatively inaccurate or imprecise measures. For example, the function point (FP) count of a project is calculated by summing up discrete values of its component functions, none of which is more granular than 3 FP. To then calculate defect density and report it with multiple decimal places leads to the mistaken conclusion that the metric is exact. The same situation arises when sophisticated estimating models produce effort estimates to fifteen minute accuracy based on input variables that may have been guesses (e.g., project risk on a 1-5 scale). We all know intuitively that

estimates based on a myriad of input variables cannot accurately predict schedules to the closest fifteen minutes (let alone the number of hours) yet I routinely encounter professionals who cite hour estimates with at least 1 decimal place (doesn't this imply that your estimate is accurate to the closest tenth of an hour or 6 minutes?)

6. **Use common sense and statistics to correlate collected data, and question figures that seem out of line.** Don't accept data purely at face value without verifying its consistency or accuracy. Many companies collect work effort data on completed projects, but the definition of project work effort can vary widely across different teams (e.g., overtime recorded/not recorded, resources included, work breakdown structure, commencement/finish points, etc). Be careful not to compare data that appears comparable because of common units (e.g., hours) that is actually based on different measurement criteria. For example, two projects may report 100 development hours, but one included overtime and user training hours while the other did not. Although the units are the same, the hours are not comparable. "Project hours" has no industry wide definition and can vary widely – ensure that your organization has established a consistent definition for collecting and reporting project hours for any projects included within the scope of data collection. Further information and tips about how to ensure consistent project effort tracking will be presented in an upcoming article.

Conclusion

These are a few of the factors, both human and technical, that can lead to software measurement success. There is a great deal to be gained by tracking and controlling software development through measurement — if only companies would consider what various measures can provide, rather than seeking a non-existent silver bullet to solve all their measurement needs.

====================

***About the Author**: Carol Dekkers is a leading software measurement practitioner and president of Quality Plus Technologies, Inc. Quality Plus Technologies,Inc. provides professional software measurement training and consulting services as well as IFPUG certified function point training, mentoring and consulting services. She can be reached at dekkers@qualityplustech.com. Note: additional articles on the subject, including how to set up measurement programs, are available by sending an e-mail to the author or accessing the article request form at www.qualityplustech.com*