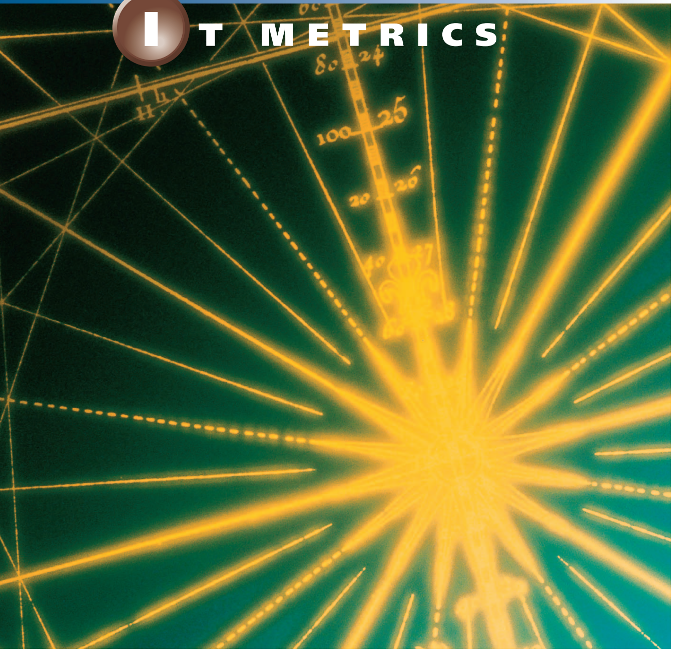


IT METRICS

Lightweight Software Metrics: The P10 Framework

William W. Agresti

All too frequently nagging questions arise in the IT workplace. Two such questions are, "Why are we still getting this report?" and "Am I supposed to make sense of these numbers?" What has happened is the metrics and reporting have drifted away from anything meaningful or actionable—or perhaps they never even started out that way.

The P10 framework guides the definition of meaningful software metrics for IT projects.

Software metrics programs must provide value. Metrics need attachment to what's important about a software project. A thorough and comprehensive way to help ensure that connectedness is the Goal Question Metric paradigm (GQM) developed by Victor Basili of the University of Maryland. GQM proceeds in an orderly fashion to consider goals and specific questions so that when the metrics are defined they relate directly to project goals.

STARTING POINT: P5

When projects don't engage in a GQM exercise, a quick-and-dirty approach has been to focus on critical P5 items: people, problem, process, products, and progress, as shown in Figure 1. Metrics that answer key questions about these P5 elements would go a long way toward providing needed visibility into a software project's critical risk and control issues. Here are the P5 items defined:

Inside

Further Reading
Web Resources

- *People* are the individuals and their roles and responsibilities on the team.
- *Problem* is the task your project is addressing, its size and scope.
- *Process* is the set of interrelated tasks, events, and milestones you follow to get the job done.
- *Products* are the deliverables in their required forms and specified quality.
- *Progress* is understanding when you are truly making headway; it is more than meeting the milestones.

The P5 framework was relevant in a world of custom, in-house development. However, as software and systems evolve (with improved methods and practices) and revolutionize (with agile processes, off-the-shelf components, and international partnering), the simple P5 model needs updating.

IMPROVED: P10

Figure 2 shows the P10 model as the framework for a lightweight metrics program—still simple and mnemonic for its prominent "P" words, and now, it is hoped, more relevant.

The supersizing of the framework to P10 has added project features that capture the emerging importance of the following:

- *Partners.* Your project is often producing products that must integrate with those of other development organizations;
- *Performance.* Your products are expected to interoperate in an operational environment (you cannot wait until the end of the project to find out if they perform as expected);
- *Plans.* Always necessary, but now more critical with often complex inter-organizational development efforts;

Published by the IEEE Computer Society

Figure 1. P5 metrics.

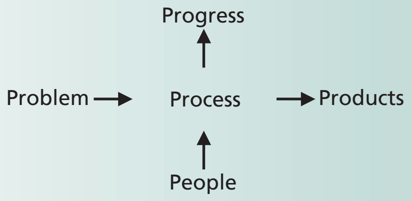
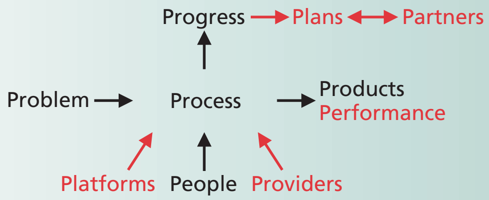


Figure 2. Improved P10 metrics.



- *Platforms.* The development and test environments must be configured properly, possess the necessary tools, and be accessible as planned to meet schedules; and
- *Providers.* Success of your project increasingly depends on expertise providers, and on components over which you have little or no control.

In associating metrics with the P10 elements, we adopt a generous definition of *metric* ("Everything is Measurable," Douglas Hubbard, *CIO Enterprise Magazine*, 15 Nov. 1997, www.hubbardresearch.com):

- Do not agonize excessively about attaining the lofty level of a "true metric" that has perfect construct validity for rarified concepts like software quality.
- Celebrate the benefits of simply being able to establish upper and lower bounds on quantities; interval-estimates more realistically express behavior of the often poorly understood phenomena we experience.
- Recognize the merits of qualitative characterizations and profiles when numbers are unavailable, cost too much to obtain, or are unreliable.
- Resist the temptation to sit on the sidelines waiting until someone comes up with true metrics that truly excite you.
- Be willing to accept mere "indicators" when they are meaningful and useful to you, even if they don't promise to capture the entire essence of an entity's property.

IMPORTANT P10 QUESTIONS

The following is a starting list of key questions and observations about the P10 software metrics. It is intended to prompt you to consider what is important to measure in each case.

Partners

Who are they? Can they be trusted? For each external partner, what is the nature and formality of the relationship? For example, which relationships are enforced by contract, memorandum of understanding, interface control documents, implied understanding, or handshakes? What are your partners' expectations for what you are doing, and how well do those expectations match what you really are doing?

People

Who is supposed to be working on this project? Who is actually working on this project? Who are the people whose contributions are critical to success—and are these people working at the level you expect? Are people trained appropriately? How does planned training compare to actual training received?

Performance

What are the performance expectations for your software product? How closely does your product test en-

Further Reading

- ➤ *The Goal/Question/Metric Method: A Practical Guide for Quality Improvement of Software Development,* **R. van Solingen and E. Berghout, McGraw-Hill, 1999.**
- ➤ *Practical Software Measurement*, **J. McGarry and colleagues, Addison-Wesley, 2001.**
- ➤ *Software Metrics: Best Practices for Successful IT Management,* **P. Goodman, Rothstein Assoc., 2004.**
- ➤ *Best Practices in Software Measurement,* **R. Dumke and colleagues, Springer, 2004.**
- ➤ *The Poor Person's Guide to Estimating Software Development Costs*, **D. Reifer, IEEE CS Press, 2006.**
- ➤ *Software Measurement and Estimation: A Practical Approach,* **L. Laird and C. Brennan, Wiley, 2006.**
- ➤ **"Software Metrics Roadmap," N. Fenton and M. Neil, http://www.cs.ucl.ac.uk/staff/A.Finkelstein/fose/finalfenton.pdf.**

Web Resources

- Hubbard Decision Research (www.hubbardresearch.com).
- Software Technology Support Center, Hill Air Force Base, Utah (www.stsc.hill.af.mil/).
- Data and Analysis Center for Software (DACS) (http://iac.dtic.mil/dacs).
- Software Engineering Institute (www.sei.cmu.edu).
- Software Program Managers Network (www.spmn. com).

vironment match its eventual operational environment? How do the results from simulations, off-line studies, test environments, and integration exercises impact your confidence that your product will integrate and perform as expected?

Plans

What is guiding all aspects of the project, so it is not an ad hoc operation? Are your plans (that is, development, test, configuration management, integration, product assurance) consistent with each other? Are they up to date? Are the plans clear on which policies and practices must be followed on the project? What is the funding plan? How realistic is the funding stream for the planned development? Who are your customers and users for this project? How much do you know about them and their expectations? If you need more money to complete the project, who decides if you get it?

Platforms

What control do you have over the development and test environments that you are counting on? Who controls changes to tools or other elements in the environments? How robust are the development and test platforms—are you prepared for contingencies? Are you tracking the actual availability and accessibility that you have?

Problem

What is it exactly that you are being asked to do? It is a striking reality that this most basic question is often the true source of late and defective products. What are the various information sources on what the job entails—are you relying on interviews, written descriptions, specifications? True, "problem" is shorthand for the more acceptable term "requirements." Can the requirements be enumerated so that it makes sense to say that you have, for instance, 134 requirements or 582 function points? How precedented is this job: Has something like this ever been done before? Is it doable with existing resources or will success require technological advances? In particular, how well do you understand the nonfunctional requirements ("Software Quality from a Behavioral Perspective," J.M. Voas and W.W. Agresti, *IT Professional*, Sept.-Oct. 2004) such as availability, maintainability, and security provisions? How is the job changing? Track the requirements changes over time. When changes are proposed, be ready to push back and explain their impact on cost and schedule. Learn more about the people you are trying to please; find out why your project is important to their goals. Who will judge project success? How often do you get to talk to them? Perhaps their expectations have changed since you started the job. Who is allowed to make changes to the requirements? What control or influence do you have over them?

Process

How will you apply people, platforms, provided elements, and other resources to address the problem? Are you following a defined software process? If so, have you identified deviations from the nominal process, and do the team and management know about them? What activities, at a useful granularity level for you, are being performed? Have you aligned payroll reporting codes and subcodes with activities in your work breakdown structure so you can get payroll to do some of the data collection for you? If your contracts or customer expectations call for milestones or process reviews that don't make sense, can you negotiate them away, in favor of ones that do make sense?

Product

Can you enumerate all the products required? For each product, what form is expected and how will you ensure high quality? Are you tracking actual quality versus required/desired quality levels? Can you deliver software early to get user reactions that help ensure final-version acceptance? Are stakeholders clear on the delivered product's capabilities? Track the precursors of eventual product quality, such as inspection reports, design reviews, and

Who sets computer industry standards?
802.11
gigabit Ethernet
firewire
Together with the IEEE Computer Society, you do.
Join a standards working group at
www.computer.org/standards/

the experiences of focus groups with user interface prototypes. What do they imply about potential future shortcomings that you can address now?

Progress

When will you be done? What does "done" mean—for you, for the project's stakeholders—and is it the same for all? What will mark real progress on the project, and are you tracking those events? For example, real progress may hinge on a critical design decision, such as committing to an operational environment or using an off-the-shelf component. Making that commitment may be the springboard for future progress. Are you tracking action-item completion and open-question resolution—issues that must be resolved to make progress? What, according to management, constitutes progress, and are you tracking that?

Providers

What are the outside (defined by you) sources of expertise or components on which your project depends? What can you do to influence or control those external entities to ensure that contributions are timely and meet your needs? Have you planned adequately for the time to integrate reusable and off-the-shelf components into your product? Do the actual scheduled arrivals of contributions—and their quality—meet your plans? How stable are the providers: what is the extent to which they can be counted on, based on the financial and other risks they face? How realistic are contingency plans in the event providers do not deliver as planned?

There is great variety in today's software creation processes. When organizations used to develop their own software, projects required tracking a simpler set of factors. In sharp contrast, software today may reflect the use of scripting languages, multimedia data, contextual frameworks (like ERP systems), international partners, open source components, dynamic linking, and specialty content providers. It is only natural that the range of factors critical for a successful software development project has grown as well.

The P10 software metrics framework better reflects the contemporary software development environment. P10 is a simple way to help ensure that the software project metrics you use will provide answers to the important questions you face in creating on-time, high-quality software. ■

William W. Agresti *is professor of information technology at Johns Hopkins University. Contact him at agresti@jhu.edu.*

For further information on this or any other computing topic, visit our Digital Library at http://www.computer.org/publications/dlib.