10 Steps to Better Software Project Metrics

Fred Brooks' observation that nine women can't make a baby in one month is perhaps the earliest and best known analogy between software development and parenting, and it's an apt one. An effective software measurement program — like good parenting — requires careful planning, regular monitoring, and a significant long-term investment of time and energy.

Unfortunately, many organizations collect reams of data that produce negligible ROI. Metrics programs that can't demonstrate tangible results are usually the first casualties when the budget belt tightens. My experience with software measurement and process improvement over the past 20 years indicates that management interest in software measurement follows a cyclical pattern:

1. Management gets bitten by the measurement "bug."
2. Money is spent to collect and analyze software project data, often with unclear goals (*"Just start collecting software project costs and other estimating/ benchmarking data"*). Little thought is given to how metrics will be used to support future decisions.
3. Data is collected and analyzed, resulting in changes to software cost estimation and project management processes. Software estimation gets better, reports are produced, and productivity appears to improve. The measurement program appears to be "on track" and life is good.
4. Often within six months of starting software measurement, something in the business changes (management or finances or competitiveness) and suddenly management questions the ROI for software measurement and other initiatives like process improvement or quality assurance.
5. Software measurement becomes the scapegoat for missed deadlines and overrun budgets*("If we weren't spending so much time collecting metrics, we could have finished on time!"*). Unrealistic expectations *("Measurement was supposed to help us manage things better!"*) prevail.
6. Management changes, budgets are scrutinized, and software measurement is abandoned, canceled, or postponed.
7. In hindsight, everyone saw the train wreck coming: (*"I knew software measurement was going to fail."*) Measurement is blamed, and becomes a dirty word.
8. Months or years pass before the organizations cycles back to step one.

The dangers and costs associated with poorly planned programs include wasted time and money, flawed decision making, loss of morale, frustrated staff, rework and competitiveness. There is a better way to create

success with measurement. Effective and successful software measurement requires careful planning and active management that anticipates and addresses dysfunctional behaviors measurement can cause. Here are 10 steps to creating and maintaining a successful software measurement program.

1. Identify the Dangers

Simple awareness that poorly planned metrics can cause damage is enough reason to follow best practices in metrics program design. Two approaches that embrace this are the GQM (Goal Question Metric) method by Victor Basili and the PSSM (Practical Software and Systems Measurement) from the Software Engineering Institute formerly of Carnegie Mellon University.

Limit data access to those who are skilled in data analysis. You wouldn't allow a child to play with matches — do not allow management access to raw or invalidated data. Proper data analysis and correlation is a critical success factor for any metrics program.

Be realistic with management about their expectations. A program designed to meet champagne tastes (or measurement results) on a beer budget seldom succeeds. Moreover, sometimes historical data can be collected if available, while at other times data are impossible to collect after the fact.

2. Make a Plan and Start Small

Project data can be collected quickly and easily, but it takes careful planning to make sure such data will be *useful*. In the same way that litter can pose dangers to infants, haphazard data collection can lead to measurement rework and failed programs. Identify a use for all data (that's the Goal and Questions part of the GQM approach) *before it is collected*. A few well-chosen metrics are better than hordes of unusable data.

QSM's Larry Putnam, Sr. recommends using five core metrics to measure the performance of software development projects: size, productivity, time (duration), effort, and reliability (Putnam, *Five Core Metrics—The Intelligence Behind Successful Software Management*). Simple metrics like this are immediately useful for helping you quantify risk on future estimates. In an industry study completed by QSM's Kate Armel, 26% of projects overran their planned budget by at least 20% — disasters that might have been avoided by collecting schedule and effort overrun data from completed projects and using it to buffer future estimates.

3. Pilot the Plan and Adjust Where Needed

Once you've identified core metrics and data to collect, test the measurement processes by doing a pilot project with a small group of participants. Run them through training, pilot the data collection, perform preliminary data analysis, and ensure that your plan and results are solid before rolling them out. In this way, flaws in the process and overall plan are identified and addressed *before* they create the need for costly damage control.

Pilot the measurement and metrics in a controlled environment before rolling the program out. Train the right people to collect and analyze the metrics to make sure the intended results are achievable. It is far easier to identify and correct undesirable behavior (often unintended consequences of measurement) in a controlled environment and minimize potential damage while the program is still small.

4. Be Honest and Open

Metrics driven process improvement is often one of the noble goals of measurement, yet companies often hide metrics planning behind closed doors. This secrecy often gives rise to rumors. Remember: "perception becomes reality in the absence of fact". If the rationale for measurement is kept a secret (even if it is to analyze

productivity levels in anticipation of outsourcing,) rumors will replace facts.

Minimize rumors and their impact by making the measurement program details open and transparent.

5. Check that the Process Is Working, then Check Again

Just as it is important to check that doors are secured and escape routes have been blocked for infants, it is important to ensure that data is being collected properly. Change and measurement seldom appear comfortable (or even desirable) to those collecting data. A few will find workarounds to avoid (or even sabotage) the data collection.

When data comes in that is too good to be true or outside reasonable control bounds, spot check adherence to the collection process to determine whether the outliers reflect real variation or faulty measurement. Put checks and balances in place to ensure the data is collected and interpreted consistently and correctly.

6. Beward of Silence

Silence often warns parents that children doing something they shouldn't (children are always loud unless they are sleeping or getting into trouble). Don't assume that no complaints mean everything's copacetic. There *will* be resistance to measurement. If you don't hear grumbling, discontent may have gone underground. It is far better to air complaints and address opposition head on than to have them undermine a program down the road.

7. Test Results Before Trusting Data to the Masses

Before you proclaim success and report on your findings, test your reports with peers or colleagues to make sure the data supports your conclusions. The worst damage often occurs when management or ill-informed (but well intentioned) people make decisions based on what they think the data is telling them.

For example, one of the most common "fatal" errors in measurement programs is to collect data and prematurely publish metrics that result in punishing of individuals. Productivity (hours per FP for example) depends on tools, techniques, language and many factors other than team skills. Faulty analysis can lead to premature or wrongheaded changes in policy or processes. It's better to test out the metrics and see what kind of causality comes up (or how the data can be misinterpreted) ahead of time before sending it out in wide distribution.

Do not allow data to get into the hands of untrained users. Often sponsors want to "play with the data" once some measurement data is collected. Avoid the temptation to release the data to anyone who simply asks for access.

Perform a dry run with any metrics reports before distribution. After data analysis is done, gather a small focus group together to evaluate the results. It is far easier to identify and correct misinterpretations of a chart with a small group than it is after a large group sees it. For example, if a productivity bar graph of projects leads viewers to the wrong conclusion, it is easy to correct the charts or add context. It only takes one wrong action based on data misinterpretation (i.e., firing a team from an unproductive project when it was actually a problem of tools) to derail a metrics program.

8. Don't Shoot the Messenger

Collected measurement data are never good or bad — they simply reflect information about a project or process as it exists today. The worse the data appear, the more opportunity exists for process improvement. But when initial productivity levels or quality data are rarely as "good" as anticipated, it can be tempting to blame the

messenger or discount the message.

When professionals follow directions, collect requested data, and submit their analysis they should be rewarded for their efforts — no matter what results the data show. Remember that measurement and metrics data report the results of a process and not of the people doing the process.

9. Be Clear and Consistent

Be honest and specific about project resources, schedules, and intended results. Condition management not to expect unreasonable results from measurement. It often takes upwards of 6 months to a year to collect and analyze enough data to support solid analysis that improves strategic decision making.

10. Accept that Curiosity and Wishful Thinking Are Unavoidable

Recognize that inflated expectations and wishful thinking will not disappear overnight. Management and staff may not understand that good measurement requires training (in metrics concepts), planning, design (measurement processes), implementation (initial data collection), training (for the program), communication, etc. As a result, people may give lip-service to the overall initiative without understanding that metrics projects require active management. Communication and goal-setting will be an ongoing process.

Remember, new software measurement programs are vulnerable. Protecting them requires active supervision and proactive risk management. Just as experienced parents cover electrical outlets, block stairs, and place sharp objects out of reach, experienced metrics managers must proactively identify risks, prevent misuse of metrics, and protect new measurement programs until they are strong enough to stand alone and prove their value to the organization.