



Key Measures for Project Management

Steven M. Woodward

INTRODUCTION

Identification of the right project measures needs to be evaluated carefully. Not enough measures and the analysis may be incomplete and misleading. Too many measures and the accuracy of the information collected by the teams may degrade, and the analysis will be more complex.

Depending on your organizations and project goals, different sets of measures with appropriate definitions will need to be established and collected. The “Goal/Question/Metric” is a popular approach to identify the measures to collect, by first establishing your projects/organization goals, then identifying specific questions, then finally the identification of the measure itself. The careful analysis of your goals, the questions which the measures need to answer, and the measures and definitions are key preliminary steps, which cannot be underestimated in their importance.

The six most common software project measures are Function Points, Effort, Cost, Schedule Duration, Defects and Lines of Code. This article will focus on these popular measures, describing their usage, strengths, challenges, and implementation.

THE MEASURES

Function Points

Definition: Function Points are a unit of measure that quantifies the volume of software functionality contained in an application or project. Just as liters are a volumetric measure of liquid, Function Points are a volumetric measure of software functionality.

Description: The International Function Point Users Group (IFPUG) provides a standard framework with guidelines to communicate and promote consistent functional sizing of any software from real time, military or telecommunication to batch financial applications using function points.

Utilization: Software functionality is what IT organizations develop, enhance and support. Therefore function points are a key unit of measure when assessing project size, establishing estimates, and managing risk.

Recommendations: Function Points should be a core measure for any software development/enhancement effort. Function Points are a well-documented standardized process supported by an internationally recognized organization, IFPUG. Their use focuses the technical team on delivering functionality that meets the users needs and, as a result, benefits the organization as a whole.

Effort Hours

Definition: Effort Hours reflect the number hours required to complete a software activity and its associated tasks. Effort Hours should be tracked by the activity and summarized for the entire project.

Description: The project effort should be captured and tracked at an appropriate level of detail to satisfy the project tracking requirements. Unlike function points, there is no international body that establishes standards for effort tracking. Your organization will need to define what effort hours will be included/excluded. For example, will unpaid overtime, vacation, training, management, administration be included or excluded from the project time.

Utilization: Effort Hour units will be used during the project to monitor progress and report productivity metrics. It will subsequently be used to help estimate future projects using the organizations historical data. It is therefore necessary that this measure be consistently and accurately collected across projects to ensure accurate performance analysis and future project estimates.

Recommendations: Accurate time accounting is critical in order to evaluate the accuracy of previous estimates and is the source for future estimates based on historical data. Specific guidelines need to be established for your organization. It is critical that the scope of the activities

to be included be agreed upon early in the project lifecycle.

Cost

Definition: Project Cost refers to the funds expended to develop and deliver the software solution.

Description: Project Cost can be extrapolated from the effort hours, by multiplying them by an agreed upon hourly rate. There may be additional costs to consider such as hardware/software, installation, and even maintenance support. Clearly document which costs are included or excluded in your project to aid in comparative analysis.

Utilization: Cost is obviously an important aspect to communicate to the project sponsor. Costs should be estimated, collected and tracked appropriately so that cost analysis can be completed to answer any potential questions.

Recommendations: Similar to Effort Hours be very specific of what is included or excluded from consideration. Costs pertaining to purchasing hardware/software, maintenance, and customization need to be agreed upon early in the project.

Project Duration Days

Definition: Project Duration Days is the number of calendar days from the start of the project at the beginning of requirements through implementation.

Description: As with effort hours, there are no industry standards that define project duration. Your organization needs to agree on what start/end dates to track, and what criteria should be used to determine when a milestone is met.

Utilization: Tracking the project duration helps the team recognize potential schedule compression early. Projects with condensed schedules (less calendar time than normal) usually have higher costs and defect rates. This is primarily due to the fact that larger than necessary project teams inhibit effective communication and coordination of project activities. Project duration is also a required measure when evaluating project progress.

Recommendations: Project duration should be tracked, using your own definitions for “when does the clock start” and “when does it end”. For example, when is the project considered completed? When is it “installed” at the first site? Or at all sites? If the rollout is at 1,000 sites this distinction is very important.

Defects

Definition: A defect is a problem or error that, uncorrected, will produce unsatisfactory results. Unsatisfactory results range from cosmetic, to an inoperable systems.

Description: These are often fondly referred to as “bugs”, “problems” or as some software development group call them, “supplemental features”. It’s just not software that can have defects; there may be defects in the requirements, design, code or documentation. Defect information typically includes the origin (source), category, and severity level (impact).

Utilizations: Defects are used to help quantify the quality of the delivered product and helps identify where and when corrective action needs to take place during the project. The information is useful to help determine “is it good enough” and forecast maintenance staffing levels. Defect-prone applications will require more support staff for maintenance support. Root cause defect analysis can also provide useful information to improve requirements definition techniques.

Recommendations: Evaluate your cultural readiness; the organizational needs to realize that finding defects before your customer is a good thing to do, not a bad one.

Lines of Code

Definition: Lines of code refers to executable LOC (Lines of Code), or may also be known as KLOC (Kilo Line of Code) for thousand lines of code.

Description: A line of code should be an “executable line of code”, not physical. Comments and blank lines should not be counted. Similarly, if an executable line is dispersed over several physical lines it should only be counted once.

Utilization: Useful to internal project teams for planning/tracking purposes, but has limited value as a project measure. I have never had a client’s end user care how many lines of code it took to deliver a business function; they want the business function. Function Points are a more appropriate measure to communicate an applications size to an end user.

Recommendations: It may be easy to get LOC when coding using a language such as COBOL or RPG, but when coding in PowerBuilder™ or VisualBasic™ it becomes more difficult. It is difficult to estimate Lines of Code until the project is near completion; therefore care should be taken when using LOC for planning/estimation/tracking purposes.

Summary

Careful planning and analysis will permit your organization to identify its key measures with appropriate definitions to help answer your organization questions for a particular project. When everyone understands “a mile” is “a mile” and “a kilometer” is “a kilometer” and we have specific definitions, which are understandable, clear

and repeatable, then the information can be used and leveraged with confidence.

About the author

Steven Woodward wrote this article. Mr. Woodward operates Q/P Management Group of Canada. He specializes in software measurement, process improvement, risk management, and software benchmarking. His areas of expertise include software measurement, estimating, quality assurance, outsourcing management, project management, function point analysis, and information services education