QUESTCON®
Technologies
A Division of Howard Systems International, Inc.

Test Metrics:
A Practical Approach to Tracking & Interpretation

Prepared By:
Shaun Bradshaw
Director of Quality Solutions
Questcon Technologies

Abstract
It is often said, "You cannot improve what you cannot measure." This paper discusses Questcon's philosophy regarding software test metrics and their ability to show objective evidence necessary to make process improvements in a development organization. When used properly, test metrics assist in the improvement of the software development process by providing pragmatic, objective evidence of process change initiatives. This paper also describes several test metrics that can be implemented, a method for creating and interpreting the metrics, and illustrates one organization's use of test metrics to prove the effectiveness of process changes.

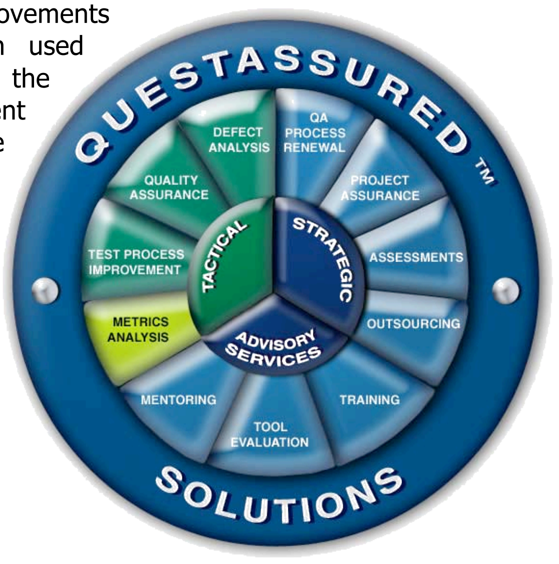

TABLE OF CONTENTS

Introduction

In 1998, Questcon Technologies assembled a special Methodology Committee to begin the process of defining, collecting, and creating a set of methodologies to be used by our consultants in instances where our clients had no existing or preferred QA or testing process. These methodologies were crafted using various sources, including industry standards, best practices, and the practical experience of the Methodology Committee members. The intent was to develop a set of practical, repeatable, and useful approaches, tasks and deliverables in each of the following areas:

- Software Testing,
- Software Test Management,
- Software Configuration Management,
- Software Test Automation, and
- Software Performance Testing.

During the development of these methodologies the Committee recognized the need to measure the effectiveness of the tasks and deliverables suggested in the various methodologies, which led to the inclusion of a Metrics section in each of the methodologies. However, as time went on we found that there was significant interest on the part of our clients in the metrics themselves. For that reason we developed a stand-alone Metrics methodology. The goal of this paper is to briefly describe the resulting Metrics methodology.

Metrics are a standard of measurement, used to gauge the effective-ness & efficiency of project activities.

Definition

Metrics are defined as "standards of measurement" and have long been used in the IT industry to indicate a method of gauging the effectiveness and efficiency of a particular activity within a project. Test metrics exist in a variety of forms. The question is not whether we should use them, but rather, which ones should we use. Simpler is almost always better. For example, it may be interesting to derive the Binomial Probability Mass Function[1] for a particular project, although it may not be practical in terms of the resources and time required to capture and analyze the data. Furthermore, the resulting information may not be meaningful or useful to the current effort of process improvement.

[1] The Binomial Probability Mass Function is used to derive the probability of having x successes out of N tries. This might be used by a QA organization to predict the number of passes (or failures) given a specific number of test cases to be executed. For further information on this metric go to: http://www.itl.nist.gov/div898/handbook/eda/section3/eda366i.htm.

One thing that makes Test Metrics unique, in a way, is that they are gathered during the test effort (towards the end of the SDLC), and can provide measurements of many different activities that have been performed throughout the project. Because of this attribute, Test Metrics can be used in conjunction with Root Cause Analysis to quantitatively track issues from points of occurrence throughout the development process. Finally, when Test Metrics data is accumulated, updated and reported on a consistent basis, it allows trend analysis to be performed on the information, which is useful in observing the effect of process changes over multiple projects.

The Questcon Test Metrics Philosophy

Keep It Simple

Track the easiest metrics early in the development of a metrics pro-gram.

Questcon's Metrics methodology begins by showing the value of tracking the easy metrics first. So, what are "easy metrics"? Most Test Analysts are required to know the number of test cases they will execute, the current state of each test case (Executed/Unexecuted, Passed/Failed/Blocked, etc.), and the time and date of execution. This is basic information that is generally tracked in some way by every Test Analyst.

When we say "keep it simple" we also mean that the Metrics should be easy to understand and objectively quantifiable. Metrics are easy to understand when they have clear, unambiguous definitions and explanations. Below is an example of the definition and explanation of a 'Blocked' test case.

Definition	Explanation
The number of distinct test cases that cannot be executed during the test effort due to an application or environmental constraint.	This metric defines the impact that known system defects are having on the test team's ability to execute the remaining test cases. A 'Blocked' test is one that cannot be executed due to a known environmental problem. A test case is also 'Blocked' when a previously discovered system defect is known to cause test failure. Because of the potential delays involved, it is useful to know how many cases cannot be tested until program fixes are received and verified.

Questcon believes it is important to both define the metric and explain why it is important and how it is used. In the example above, the definition and explanation clearly indicated that the metric is intended to track unintentional blocks. It eliminates confusion for the Test Analyst as to when to set a test case to a 'Blocked' status.

Furthermore, this metric is objectively quantifiable. Based on the definition, a test case can be counted as 'Blocked' or not, but not both; ensuring that a blocked test case cannot be placed in multiple status “buckets”. Finally, by sticking to the definition and explanation, a Test Analyst cannot apply some other subjective criteria that would allow a test case to be flagged as 'Blocked'.

Ensure that all metrics are meaningful to your organization by clearly defining them and ensuring they are used to achieve an established goal.

Create Meaningful Metrics

Test Metrics are meaningful if they provide objective feedback to the Project Team regarding any of the development processes - from analysis, to coding, to testing – and support a project goal. If a metric does not support a project goal, then there is no reason to track it – it is meaningless to the organization. Tracking meaningless metrics wastes time and does little to improve the development process.

Metrics should also be objective. As indicated in the sample definition shown in the previous section, an objective metric can only be tracked one way, no matter who is doing the tracking. This prevents the data from being corrupted and makes it easier for the project team to trust the information and analysis resulting from the metrics. While it would be best if all metrics were objective, this may be an unrealistic expectation. The problem is that subjective metrics can be difficult to track and interpret on a consistent basis, and team members may not trust them. Without trust, objectivity, and solid reasoning, which is provided by the Test Metrics, it is difficult to implement process changes.

Automate the process by which metrics are tracked. This reduces the amount of time required for collection and the possibility of introducing human error or bias.

Track the Metrics

Tracking Test Metrics throughout the test effort is extremely important because it allows the Project Team to see developing trends, and provides a historical perspective at the end of the project. Tracking metrics requires effort, but that effort can be minimized through the simple automation of the Run Log (by using a spreadsheet or a database) or through customized reports from a test management or defect tracking system. This underscores the 'Keep It Simple' part of the philosophy, in that metrics should be simple to track, and simple to understand. The process of tracking test metrics should not create a burden on the Test Team or Test Lead; otherwise it is likely that the metrics will not be tracked and valuable information will be lost. Furthermore, by automating the process by which the metrics are tracked it is less likely that human error or bias can be introduced into the metrics.

Use Metrics to Manage the Project

It is not enough to have a set of metrics that are tracked on a regular basis. The metrics must also be reviewed and analyzed regularly, as they can provide value feedback during and after a software development project.

Many times, members of a Project Team intuitively understand that changes in the development lifecycle could improve the quality and reduce the cost of a project, but they are unwilling to implement changes without objective proof of where the changes should occur. By meeting on a regular basis throughout a project, but especially at the end of a project, the team can review the Test Metrics and other available information to determine what improvements might be made. Here is an example of how metrics can be used to make process changes during a test effort.

> Imagine that the Test team is halfway through the test execution phase of a project, and the Project Team is reviewing the existing metrics. One metric stands out - less than 50% of the test cases have been executed. The Project Manager is concerned that half of the testing time has elapsed, but less than half the tests are completed. Initially this looks bad for the test team, but the Test Lead points out that 30% of the test cases are 'Blocked'. The Test Lead explains that one failure in a particular module is preventing the Test Team from executing the 'Blocked' test cases. Moreover, the next test release is scheduled in 4 days and none of the 'Blocked' test cases can be executed until then. At this point, using objective information available from the metrics, the Project Manager is able to make a decision to push an interim release to the Test Team with a fix for the problem that is causing so many of the test cases to be 'Blocked'.

As seen by this example, metrics can provide the information necessary to make critical decisions that result in saving time and money and makes it easier for a project team to move from a "blame" culture to a "results-oriented" culture.

Base metrics are the raw data gathered by a Test Analyst during a test effort and are made up of basic counts of data elements.

Types of Metrics

Base Metrics

Base metrics constitute the raw data gathered by a Test Analyst throughout the testing effort. These metrics are used to provide project status reports to the Test Lead and Project Manager; they also feed into the formulas used to derive Calculated Metrics. Questcon suggests that every project should track the following Test Metrics:

# of Test Cases	# of First Run Failures
# of Test Cases Executed	Total Executions
# of Test Cases Passed	Total Passes
# of Test Cases Failed	Total Failures
# of Test Cases Under Investigation	Test Case Execution Time*
# of Test Cases Blocked	Test Execution Time*
# of Test Cases Re-executed	

As seen in the 'Keep It Simple' section, many of the Base Metrics are simple counts that most Test Analysts already track in one form or another. While there are other Base Metrics that could be tracked, Questcon believes this list is sufficient for most Test Teams that are starting a Test Metrics program.

Calculated Metrics

Calculated metrics use the base data to develop more useful information for doing metrics analysis and identifying potential SDLC process improvements.

Calculated Metrics convert the Base Metrics data into more useful information. These types of metrics are generally the responsibility of the Test Lead and can be tracked at many different levels (by module, tester, or project). The following Calculated Metrics are recommended for implementation in all test efforts:

% Complete	% Defects Corrected
% Test Coverage	% Rework
% Test Cases Passed	% Test Effectiveness
% Test Cases Blocked	% Test Efficiency
1st Run Fail Rate	Defect Discovery Rate
Overall Fail Rate	Defect Removal Cost

These metrics provide valuable information that, when used and interpreted, oftentimes leads to significant improvements in the overall SDLC. For example, the 1st Run Fail Rate, as defined in the Questcon Metrics Methodology, indicates how clean the code is when it is delivered to the Test Team. If this metric has a high value, it may be indicative of a lack of unit testing or code peer review during the coding phase. With this information, as well as any other relevant information available to the Project Team, the Project Team may decide to institute some preventative QA techniques that they believe will improve the process. Of course, in the next project, when the metric is observed it should be noted how it has trended to see if the process change was in fact an improvement.

The Final Step - Interpretation and Change

Use all available information when interpreting the metrics gathered, including other metrics and intangible information that may not be quantifiable. Also, when available, compare the metrics across several projects.

As mentioned earlier, test metrics should be reviewed and interpreted on a regular basis throughout the test effort and particularly after the application is released into production. During the review meetings, the Project Team should closely examine ALL available data, and use that information to determine the root cause of identified problems. It is important to look at several of the Base Metrics and Calculated Metrics in conjunction with one another, as this will allow the Project Team to have a clearer picture of what took place during the test effort.

If metrics have been gathered across several projects, then a comparison should be done between the results of the current project and the average or baseline results from the other projects. This makes trends across the projects easy to see, particularly when development process changes are being implemented. Always take note to determine if the current metrics are typical of software projects in your organization. If not, observe if the change is positive or negative, and then follow up by doing a root cause analysis to ascertain the reason for the change.

Metrics Case Study

The establishment of a metrics program for one organization led to development process changes resulting in a reduction of rework costs totaling more than $170,000 for one three-month project. A savings that continues with each subsequent project.

Background

A recent client, the IT department of a major truck manufacturer, had little or no testing in its IT projects. The company's projects were primarily maintenance related and operated in a COBOL / CICS / Mainframe environment. The company had a desire to migrate to more up-to-date technologies for some upcoming new development projects and felt that testing involvement should accompany this technological shift. Another consulting firm was brought in to provide development resources experienced in the new technologies and Questcon was hired to help establish a testing process, and aid in the acquisition and training of new test team members.

Course of Action

As a first step, Questcon introduced the Test, Test Management, and Metrics Methodologies to the new test team. The team's first project, 'Project V', was primarily developed in Visual Basic and HTML, and was accessed via standard web browser technologies. By the time the Test Team became involved in the project, all of the analysis and most of the development (approximately 70%) had been completed. The test team tracked test metrics throughout the test effort and noted the following key metrics data:

- Developed 355 test scenarios,
- Had a 30.7% 1st Run Fail Rate, and
- Had an Overall Failure Rate of 31.4%.

Using the information available from the analysis of the test metrics, Questcon suggested that earlier Test Team involvement, in the form of requirements walkthroughs and design reviews, would improve the failure rate. Working with the same project team from 'Project V', Questcon convinced the Project Manager to institute these changes for the second project, 'Project T". While 'Project T' was more complex than 'Project V' (it added XML to the new development environment of Visual Basic and HTML) the overall size of the project was very similar, requiring 345 test scenarios.

Results

The 1st Run Failure Rate for 'Project T' was only 17.9% and the Overall Fail Rate was 18.0%, dramatically better than the results from 'Project V'. Given that the project team remained the same between each of the projects and the fact that team was already highly skilled in the technologies being used, our root cause analysis indicated that the addition of the requirements walkthroughs and design reviews were the primary factor in reducing the number of defects delivered to test. Furthermore, when costs were reviewed for each of the projects it was noted that the accumulated costs of rework were also reduced in the amount of just over $170,000.

While metrics themselves do not create improvements, they do provide the objective information necessary to understand what changes are necessary and when they are working.

Conclusion

There are several key factors in establishing a metrics program, beginning with determining the goal for developing and tracking metrics, followed by the identification and definition of useful metrics to be tracked, and ending with sufficient analysis interpretation of the resulting data to be able to make changes to the software development lifecycle. This paper has laid out Questcon's philosophy in regards to those key factors. Furthermore, we have demonstrated, through the use of the Business Case, that while the test metrics themselves do not create improvements, the objective information that they provide can be analyzed – leading to reductions in the number of defects created and the cost of rework that would normally result from those defects.

About Questcon

Questcon Technologies is a Quality Assurance (QA) and software testing company, headquartered in Greensboro, NC that has focused on researching, developing, and delivering innovative quality solutions for its clients since 1991. Utilizing its proprietary QuestAssured™ methodologies, Questcon furnishes customized QA and testing best practices that are customized to each client's processes and business objectives. We have helped hundreds of companies, from large government and private sector enterprises to smaller software organizations establish QA, development, and testing standards.