"Our testing costs are clearly too high; how can we prove this to management and determine a course of action to improve our efficiency"

January 2017

Scope of this Report

Testing is an essential part of software development, and is even more critical with the arrival of complex integrated systems with business transactions that need to be bulletproof to defects. It doesn't really matter whether you do agile or classic waterfall, testing is still going to be a big part of your lifecycle, and so it is imperative that it be as efficient as possible without reducing the expected quality. But there is another variable to this equation, and that is the cost of testing which has become a concern for many organizations. This report is divided into two parts. The first focuses on cost tracking and capturing other variables that help us make decisions regarding our testing strategy. The second analyzes several options to improve our testing efficiency.

Testing management, tracking and reporting

Testing management

It is important to define the roles and activities people have to fulfill in our testing process. We could do that through the Test Management Office (TMO) or by dividing those roles across the organization.

A TMO is especially useful when we work in an environment with several vendors and multiple applications in which each of them are using different tools, methodologies and even lifecycles. The TMO should be in control of the overall testing strategy, the quality management, the tracking and the reporting.

Benchmarking

It is a good idea to start by measuring ourselves relative to other similar companies in the market by performing a benchmark. Benchmarking gives us an idea of the overall performance levels of similar companies and other useful parameters that can be translated to our company.

However, we have to be aware that this is just a way to position ourselves. To do so, we need a reliable source of information and the necessary technical skills to perform the benchmark, or alternatively a company that can do the proper benchmark for us.

Tracking testing metrics

Testing metrics are defined as the units used to measure the parameters of our projects / portfolio with respect to testing. Those parameters could be size, quality, cost, status, etc.

270 West Lancaster Ave., Liberty Square B-2, Malvern, PA 19355 • 1-610-644-2856
info@softwarevalue.com • www.softwarevalue.com

There are a number of ways to define those metrics but first we need a procedure in place that will ensure a proper definition and use of those metrics. One example of metric lifecycle is the following:

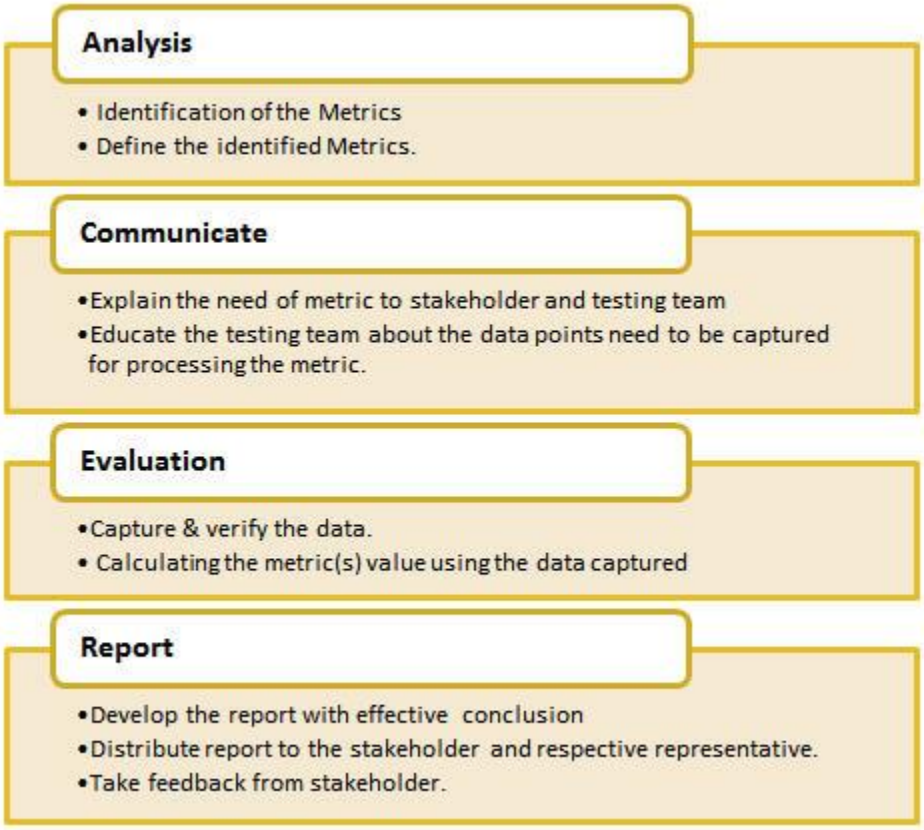

Figure 1: Metrics Life Cycle (Source: Important Software Test Metrics and Measurements)

There are two types of metrics we need to obtain through the testing process: the base metrics and the derived metrics. The base metrics are just the basic parameters that we need to gather to be able to calculate the derived metrics, which are used to formulate the reports to enable the decision-making process.

Some examples of testing base metrics are:

- Number of requirements
- Number of requirements covered
- Number of test cases
- Size of requirements
- Total number of test cases per requirement
- Total number of defects identified
- Number of test cases passed
- Number of defects fixed
- Effort expended on testing

Once we have defined and collected our base metrics, we can calculate our derived metrics to help us track cost, quality, efficiency, status, productivity and other key factors.

Some examples of derived testing metrics are:

Project Management Metrics	Metrics in Requirement & Test Design Phase	Metrics in Test Execution Phase
Effort Variation	Test Case Preparation Productivity	Test Effectiveness %
Schedule Variation	Test Design Coverage	Test Execution Coverage
Duration Variation	Review Efficiency	Test Execution productivity
Size Variation	Requirement Stability Index	Test Execution Status
Load Factor	Requirement leakage Index	Error Discovery Rate
Work Efficiency Index		Application Defect Density
Cost of Quality		Quality of fixes
Rework Effort %		Defect Leakage %
% Effort for Project Management		Defect detection efficiency
Risk Identification Efficiency %		Average defect age
Risk Mitigation Efficiency %		

Figure 2: Key metrics for a testing project (Source: Test metrics and KPIs, UST Global)

To highlight a few indicators that relate to cost we are going to detail those indicators as an example of what they represent, how they are calculated and what benefit they offer.

1. **Cost of finding a defect in testing (COFDT)**
   This indicator highlights the cost (expressed in effort) that we incurred to find a defect during our testing life cycle.

   COFDT = Total effort spent on testing / defects found on testing

2. **Time to fix a defect (TFD)**
   This indicator provides a view of the maintainability of the product that can be used to estimate project maintenance costs. It symbolizes the effort required to resolve a defect (diagnosis and correction)

   TFD = Total effort in diagnosis and correction / number of defects solved on that period

3. **Cost of quality (COQ)**

   This indicator is used to calculate the total effort spent for QA in a project. The objective is to measure the effort spent on reviews, testing and rework against the production effort.

   It consists of three types of cost:

   - Prevention Cost – The effort to ensure product quality (defect prevention. Metrics collection and analysis, training, process improvement)
   - Appraisal Cost – Quality failures detected prior and after the product shipment (rework and retesting due to internal and external failures)

- Failure Cost – Discovering the condition of the product (reviews, testing and product quality audits)

$$COQ = \frac{(\text{Effort Prevention} + \text{Effort Appraisal} + \text{Effort Failure})}{(\text{Effort Prevention} + \text{Effort Appraisal} + \text{Effort Failure} + \text{Effort on Production}) * 100}$$

Reporting

To answer the question stated in the report we should be able to find the metrics that best reflect the cost of our testing strategy. Those should reflect whether our testing costs are too high or not and why is that happening if that's the case. We then generate the necessary reports in order to show this to management.

A typical generic report that could be used in our organization is the coverage burn down chart. It is a great tool for visualization at every organizational level. It gives information as to how our project / portfolio is progressing, where did we focused most of our efforts, how much time and effort is left to achieve our objectives, etc.

It is also helpful to have a unified repository with all the data that we gathered so everyone can be aligned and share the same parameters across the organization.

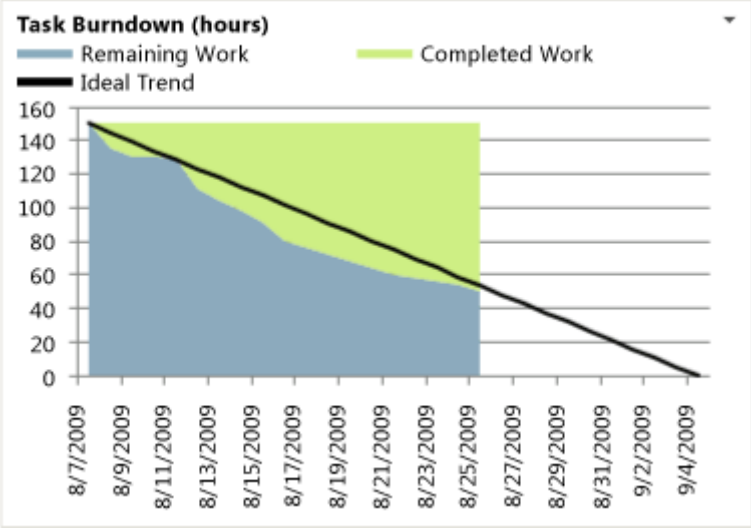

Figure 3: Burndown Report (Source: Excel reports, Microsoft)

How to improve testing efficiency?

Analyze and improve the information gathered:

Once we are able to gather and record the necessary data regarding the testing phase, the first step to improve testing efficiency is to analyze derived metrics and identify any actions to be taken.

This phase is as important as gathering the appropriate data, since the way we react to those derived metrics will shape the future efficiency, quality and cost of our testing process.

It is also important to recognize the opportunity here for continuous improvement, meaning a time to examine and adjust the data that we decide to collect; it is possible that we could find new information that could be of interest to track in the future.

Test Automation:

One of the main questions we have to answer is whether or not we want to take the time and effort to implement test automation in our organization, even though in some environments, such as continuous integration or agile, it's a given that you need to use test automation. There is no doubt that test automation helps to improve efficiency. While there are a number of benefits associated with the implementation of test automation, there are challenges that need to be mitigated and resolved. Together with the rest of the information gathered we should have enough data to make an informed decision.

Benefits:

- Return on Investment (ROI)
- Test coverage improvement
- Reduction of manual intervention (reduces errors and improves accuracy)
- Find bugs at an early stage
- Reusability
- Continuity
- Special tests (Performance, stress)
- More reliable

Challenges:

- Elevated early cost
- Appropriate tool required
- Needs to be properly balanced with manual testing (Fix specific problems from users)
- Things not testable (usability, etc.)
- Need for validation
- Communication (different teams)
- Keep things up to date (reusability needs work)
- Roles with proper technical skills

Other techniques / methodologies:

There are a number of initiatives that could potentially help us improve our testing efficiency, but it really depends on how mature our test process is as well as to whether or not we want go a step further. As reference, a couple of examples of those techniques are:

- The POMBA methodology[1]

  This is an assessment methodology, called process-oriented metrics-based assessment (POMBA). The idea is that a test process should be continuously measured in order to improve, and to do so, a set of relevant measures must be developed to provide consistent visibility to a complete process for continuous monitoring and assessment, which should, in the end, improve the cost and efficiency.

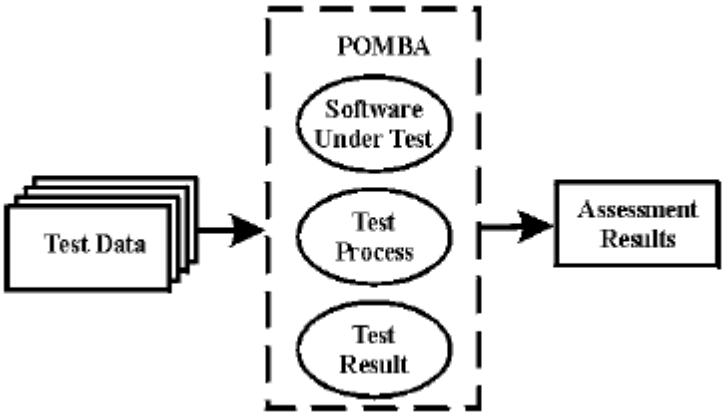

Figure 4: The top-level view of POMBA (Source: Measuring and Assessing Software Test Processes Using Test Data)

- Error-Prone Path Identification and Genetic Algorithms [2]

  This method optimizes the software testing efficiency by identifying the most error prone paths applying length generic algorithm techniques; which helps refining the effort and cost estimation in the testing phase.

Conclusion

Testing is not an easy task to manage, nor is it inexpensive but it is key to measure and manage it proactively in a software development environment. If the objective is to get control over our testing process, there are a number of things that have to be done. Firstly, we should be able to know where we are and which indicators we should track to have an overall idea of how our testing process is performing. Then there are a series of initiatives that can be performed to improve the testing efficiency

Sources:

[1] http://www.reliability-safety-software.com/wp-content/uploads/2015/03/y2ktest.pdf

[2] http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.333.3927&rep=rep1&type=pdf

http://www.getzephyr.com/resources/whitepapers/qa-metrics-value-testing-metrics-within-software-development

https://www.corpsite.ust-global.com/en/images/stories/pdf/Test_Metrics_and%20KPI_s.pdf

http://www.uploads.pnsqc.org/2012/papers/t-12_Sachdeva_paper.pdf

http://www.softwaretestinggenius.com/download/EMFST.pdf

http://www.softwaretestinghelp.com/software-test-metrics-and-measurements/
https://pdfs.semanticscholar.org/5eac/e82c2e2c7eb77af5d1f85f9fbd5ad71f67c3.pdf
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.333.3927&rep=rep1&type=pdf
http://www.jatit.org/volumes/Vol92No1/4Vol92No1.pdf