An Introduction to Functional Size and Function Points

Function points are a measure of the functional size provided to the user by an application. The user is any entity (either a human or another application), outside of the application being measured, that considers the function to be important. Size is determined by applying one of several rule sets, including those outlined by the International Function Point Users Group (IFPUG) or the Common Software Measurement International Consortium (COSMIC). It is important to note that these systems do not measure size in terms of the number of lines of code, but instead based on what the user can ultimately see and use.

Sizing software helps to establish an appropriate timeline for completing the work, as well as to appropriately allocate resources, including both team members and budget. But how does this relate to the testing process? The adage, “What you can’t measure, you can’t manage,” applies here.

As a preventative measure, it can be difficult to determine the importance of software testing or how much effort should be applied to it. Coding, by contrast, reveals its value more readily, as it provides clear-cut value by fulfilling requirements. This difference in visibility can cause testing to be less emphasized than other steps, when it should ideally be emphasized throughout the development process. What is needed is a method of making the importance and effectiveness of testing more visible and measurable. Function points can help with this, providing measures that can be applied in test planning and in measuring the effectiveness of testing.

First Step: Estimating Testing Requirements

One of the measures that is of particular use in testing is test case coverage, which is defined as:

\[
\frac{\text{Number of test cases}}{\text{Total number of function points}}
\]

This provides a measure of how thorough the testing process is for a given project. If the test case coverage from a similar project that was determined to have been successful (perhaps using a measure like defect density, described in the next section) is available, this can be used to give an estimate of test cases needed on a new project. Once the number of function points is known, simply use the number of test cases that gives similar coverage. Since function points can be measured based on requirements, rather than waiting until coding, a good idea of testing needs becomes apparent early in the process. This can be especially helpful when using defect prevention techniques (done throughout development), rather than defect removal (traditionally done at the end of development).

Initially, a software development organization may not have the data necessary to determine exactly how much testing is necessary for a given project. This does not have to be an impediment to using measurement to get an initial estimate. There are a number of sites that publish benchmarking statistics, such as the International Software Benchmarking Standards Group (www.isbsg.org), which can be used in creating initial plans. If industry data is not available for a similar project, then it is still possible for experienced testers to give an estimate based on the expected size of the final product. Once the project is finished, continued measurement can be used to help in test planning for future projects, as well as to determine if the project met expectations.

Measuring Quality

Another good measure used in testing, and an indicator of both testing effectiveness and software quality, is defect density, defined as:

\[
\frac{\text{Total number of defects}}{\text{Total function points}}
\]

I. Function Point Analysis, David Garmus and David Herron, pages 37-39
Defect density can also be compared to industry data or projects previously created to determine if the software meets standards. It does so in a neutral fashion, unlike some measures, such as cost per defect, which can penalize efforts at software quality\textsuperscript{ii}. Ultimately, testing should be part of the process of improving quality, and measuring its effects on defect density is one way to track changes in quality.

**Maintenance and Process Improvement**

Once the software development project is finished, there is still the process of maintaining it. Even during this time, software may require re-engineering at some point. The choice to re-engineer, like most business decisions, should be made at a point when it is likely to be most cost effective. Function points can be used in this decision as well. The amount of maintenance an application requires per function point can give an idea of when re-engineering should take place: the longer spent on an application that provides a given amount of benefit (measured by function points), the more likely that it needs to be re-worked\textsuperscript{iii}. This should be based on company standards or industry standards, as appropriate for the project.

Measures are also necessary for the continuous improvement of processes. While the data may not be available at the beginning of the measurement process, perhaps because industry data including a similar project does not exist, as data is collected and analyzed it can be used to improve future projects. This data collection is a requirement of certain testing models, such as the Testing Maturity Model integration (TMMi)\textsuperscript{iv}. Level Four of the TMMi model, Measured, requires test measurement, so function points and the analysis that can be based on them are one way to fulfill that need.

**Conclusion**

Software testing can be a challenge to track in a rigorous fashion. It does not itself create easily seen benefits, but instead usually prevents issues down the line. By combining function point sizings and other information, the very real benefits of testing can be more easily seen, analyzed and compared with other projects. This makes the sizing potentially valuable, allowing testing to be planned earlier in the process, giving a measure of effectiveness and allowing maintenance planning. By making the test process measurable, function points can facilitate process improvement, such as in the TMMi model. Perhaps most importantly, function points can be used to demonstrate the importance of testing to non-testers, which often includes managers. That, by itself, may be a sufficient reason to use them.

**Further Reading**


**Contact**

DCG Software Value is a global provider of software analytics, software quality management and Agile development solutions. DCG maintains a U.S. corporate office in Malvern, P.A. and a European corporate office in the U.K.

[www.softwarevalue.com](http://www.softwarevalue.com)
610-644-2856
info@softwarevalue.com

\textsuperscript{ii} Function Points as a Universal Software Metric, Capers Jones
\textsuperscript{iii} Function Points as a Universal Software Metric, Capers Jones
\textsuperscript{iv} Using Function Points, [http://www.softwaremetrics.com/Articles/using.htm](http://www.softwaremetrics.com/Articles/using.htm)
\textsuperscript{v} Test Maturity Model integration, Erik van Veenendaal and Brian Wells