Function Points or Use Case Points?

by Mauricio Aguiar

At some point in their careers many Software Measurement professionals will be asked a typical question: Do Function Points work with object-oriented software-intensive systems, or should one consider Use Case Points instead?

Software Size Measures

Software size measures first appeared as the main input to software development effort estimation. Software development effort, usually measured in staff-hours, is known to hold significant correlation with software size. The first leading size measure was SLOC – the number of Source Lines of Code. There are several different ways of counting SLOC, some of them line-oriented and some statement-oriented. SLOC is considered a physical size measure because it measures the physical volume of source code associated with a software system.

While the SLOC measure is useful in many contexts its limitations led to the appearance of other measures. Those new measures sought to measure the functionality delivered to the user by the software system as opposed to its physical size. They are therefore called functional size measures. Functional size is used to obtain early project estimates when it can be very difficult to estimate SLOC. The most important of those measures was introduced by Allan Albrecht in 1979 – Function Points. Later, other functional size measures were proposed such as *Bang*, *Mark II*, *Full Function Points*, and *Cosmic-FFP*. All those measures achieved some degree of industry use with the possible exception of Bang. In 1993 Gustav Karner created a function point variety specifically designed to measure functionality based on use cases. Use Case Points (UCPs) had been born.

Function Points

The Function Point measure originally conceived by Albrecht received increased popularity with the inception of the *International Function Point Users Group* (IFPUG) in 1986. In 2002 IFPUG Function Points became an international ISO standard – ISO/IEC 20926.

Function Points may be easily counted or estimated from use cases. Several organizations are successfully using that method as I write these lines. Pre-requisites to Function Point counting from use cases are: knowing how to read use cases and experience in Function Point counting.

Many organizations have invested significant time and money to collect data and compose large project databases containing function point data. No other functional measure has reached the same level of use and/or investment. For example, the Australia-based *International Software Benchmarking Standards Group* (ISBSG) keeps a database currently with over 4,000 projects, most of them measured in function points.

For a long time the U.S. Department of Defense and its suppliers used SLOC as the single measure of software size. That measure seemed to be adequate for a stable ADA-based environment (ADA is a language still used by the military). Around the turn of the century, research communities such as the U.S. Army PSM initiative and the USC COCOMO development group started to consider Function Points as an alternative since the appearance of new technologies made the old SLOC measure incapable of satisfying all measurement needs.

Use Case Points

Use Case Points were created by Gustav Karner in 1993 as a particular variety of function points specifically designed for use cases. Karner later went to work for Rational but that apparently did not help to make the method popular. A search for "use case points" was executed on the Rational website for the first version of this article in November 2002. It returned only one document, while a search for "use cases" on the same site retrieved 348 documents. The search was repeated in May 2009 on the IBM developerWorks website – this time 8 "use case points" documents were found while 1,619 were found for "use cases."

Even though still not very widely known, UCPs have been studied by several researchers both in industry and academia. A 2001 paper by Professor Bente Anda from the University of Oslo reported results of the application of UCP to project effort estimation. While Anda concludes that UCP can be used for estimation, his report and others suggest that use case style variations can have an impact on the number of UCPs obtained through the method.

Function Points and Use Case Points

UCP counts may vary among organizations and individuals due to variations in use case styles. It is then reasonable to assume that the productivity associated with the development of one UCP (20 staff-hours according to Karner's original work) will vary as well. Therefore to obtain reliable effort estimates one would need both to standardize use case writing styles and calibrate a local UCP-based estimation model.

The lack of universal standards for use case construction will limit comparison among projects from different organizations using UCP. There is no way to guarantee UCPs from different organizations will measure the same thing if use case writing styles are allowed to vary widely.

Furthermore, UCPs can only be used by organizations that adopt use cases to model functional requirements. That may limit comparison among companies using different requirements modeling techniques and artifacts, or even between projects from the same company before and after use case adoption.

Even though the software community has increased its UCP awareness, UCPs are still not very well known if compared to FPs. In November 2002 Google reported approximately 12,700 occurrences of “function points.” The same query for “use case points” returned only 213 occurrences. This means Function Points were approximately 60 times more cited than Use Case Points at the time. In May 2009 Google reported approximately 113,000 occurrences of “function points” as opposed to 11,400 of “use case points.” This means that six years later Function Points are still roughly 10 times as cited as Use Case Points.

Finally, there is scarcely any publicly available UCP data. That makes comparisons difficult or even impossible.

Recommendations

From an objective perspective one cannot safely recommend UCPs as the best choice for companies. Function Points are generally more convenient for the following reasons:

- Function points are maintained by a not-for-profit, international organization - The International Function Point Users Group – IFPUG, since 1986;
- Function points are supported by several consulting companies and user groups in various countries;
- IFPUG keeps a worldwide, automated certification program that ensures that certified specialists consistently apply the method;
- Function Points are an ISO standard (ISO/IEC 20926) – that helps to guarantee the uniform application of the technique;
- Function Points model requirements at a higher abstraction level than UCP. They are also artifact independent and may be used by organizations independently of the way they model requirements – they may utilize use cases or any other method;
- Studies and comparisons are made possible by the existence of a large volume of Function Point-based data kept by several organizations;
- Function Points are successfully used in contracts in government and industry in several countries.

There is nothing wrong with using UCPs as an alternative measure in addition to Function Points, perhaps to make comparisons or acquire additional knowledge. However, its lack of maturity severely limits UCP application to business relationships.

Related Publications

Albrecht, A.J. e J. Gaffney, “Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation” – IEEE Transactions on Software Engineering, SE-9, 6, 1983.

Anda, Bente, “Comparing Effort Estimates Based on Use Case Points with Expert Estimates”, Empirical Assessment in Software Engineering (EASE 2002), Keele, UK, April 8-10, 2002.

“Estimating Software Development Effort Based on Use Cases – Experiences from Industry”, In: 4th International Conference on the Unified Modeling Language (UML2001), Gogolla, M. and Kobryn, C. (editors), Toronto, Canada, October 1-5, 2001, pp. 487-502, LNCS 2185, Springer-Verlag, 2001.

Clemmons, Roy K. – “Project Estimation with Use Case Points”, Crosstalk, February 2006.

Collaris, R. and Dekker, E. – “Software Cost Estimation Using Use Case Points: Getting Use Case Transactions Straight”, The Rational Edge, 15 March 2009.

DeMarco, Tom, “Controlling Software Projects: Management, Measurement & Estimation” – Yourdon Press, 1982.

Dunn, Adam – “Function Points, Functional Requirements, Functional Documentation” – IFPUG Annual Conference, 2002.