Introduction to Software Measurement

Computer Measurement Laboratory
info@cmlab.biz

The picture below shows a simple wooden ruler. This is a simple measurement tool. We learned how to use it in first grade. Learning to use it was not an easy thing. Our first inclination was to use it as a tool to whack the girl in pigtails that sat in front of us. As we learned to count, the teacher then pointed out that there were numbers on this ruler and that the ruler could be used to quantify size attributes of things in our environment.

Later on, in our educational career, we learned that our ruler was a copy of one maintained by a standards agency. In our case this was the National Institute of Standards and Technology.

These are the two facts that we needed to understand the basic notions of measurement that we could then apply to a host of new measurement tools. First, there is a measurement tool. Then, there is a standard or reference model for that measurement tool. There are tools for measuring distance, volume, density, and many other things.

We were then taught to use our new measurement tools in the pursuit of science. In the laboratory we could measure the volume of chemicals or frogs. We could measure the amount of ammonium nitrate that we added to fuel oil to make bombs. In short, we were well on the way to becoming real scientists.

Now, the interesting thing here was that there was absolutely no leakage from our training in measurement into the new field of software development. We will not call this field computer science because the one thing that it isn't is a science. At the core of the disciplines of science is the basic notion of conduct of inquiry. New ideas are postulated and then the scientists trundle off to their laboratories to validate or refute the new ideas. In the laboratory there are measurement tools that are use in this validation process. In fact, one of the world's most expensive measurement tools has just come on line in the form of the Large Hadron Collider in Switzerland.

Oddly enough, there are no measurement tools for the field of Computer Science or Software Engineering, for that matter. There are, as a result, no laboratories and no experiments. In fact, when we look under the hood, we find that Computer Science is not a science at all. It has all of the trappings of a religion. There is dogma: OO design is marvelous. And there are priests (but it would not do to name any). There are also a host of villains or devils. These are the hackers. There is a continuing struggle between the forces of good (the priests) and evil (the devils). This whole scenario of software craftsmanship is quite reminiscent of the state of the art during the Dark Ages.

We were incorrectly taught in school that the steam engine ushered in the industrial revolution. The real "cause" of the industrial revolution was refinements in measurement tools. Steam engines must be built to very close tolerances. The parts for these engines must also be interchangeable. This is only possible with precision measure tools employed in the manufacturing process. Measurement was, in fact, the real driver in the industrial revolution. Therefore, it is entirely appropriate that software measurement be a driver in the software revolution. Imagine this new future. Programs will come with guarantees. Computer security issues will vanish. Software engineering will have real meaning in an engineering context.

Just as we learned to use a very simple measure tool at the beginning of our incipient career as nascent scientists, let us now take that simple step for software development. We will begin with a simple measure of program size, Lines of Code or LOC. This is a relative common measure of program size. The problem is, nobody knows what it is.

One of the first things that we learned about our ruler was that it was a copy of a standard ruler maintained by NIST. We now wish to build a ruler that will measure LOC for our C programs (for we know better than to avoid the pitfalls of C++ and its derivatives). To build this new ruler we will turn to NIST as we learned to do as children. They will tell us how to measure LOC. Imagine our disappointment! Not only is there no standard for measuring LOC at NIST, there are no standards for any software measurements at NIST.

Absent this very simple measure of a single program attribute, it is clear that there can be no communication between any two researchers on this subject. I can tell you that I have a C program that is 20,000 LOC. Without a standard, you will not know what this means. Given the fact that there are no standards at NIST for software measurement, it is clear that there can be no real communication among researchers in this field. Most tragically, there can be no science in computer science. So, let us take the first step to lift the field of computing software out its own Dark Ages.

A Standard for Measuring LOC in C Programs

There are many different C compilers available today. We will formulate a standard for measuring GCC C code.

For measurement purposes a C program module may not contain any compiler directives. To this end, before we measure a C program we must first remove these directives. We will use the C preprocessor CPP to do this. We will use the output of the CPP program as input to our measurement tool.

We will now read the output of the CPP program. We will ignore all lines that begin with the '#' character. Will also ignore all lines than contain only the new line character '\n'. For every other line we will increment our LOC counter until we encounter an end of file.

This will count all of the LOC for all of the modules in a file.

If it is necessary to count LOC for the program modules (functions) within a file, we must first tally LOC for all lines that are outside of any program module. Call this value LOC'. Next we will enumerate the LOC for each module. To this value, for each module, we will then add LOC'.

Now that we have a measurement standard, we can now build a measurement tool (ruler) that will implement this new measurement.

Let science begin.