Towards Well-Defined, Shareable Product Data

Warren Harrison

PSU Center for Software Quality Research
Portland State University
Portland, OR 97207-0751

1 Introduction

A major problem in experimental software engineering is the collection, maintenance and use of data. Data, especially industrial data, is painfully hard to find. Even when it is available, both its reliability and its consistency is in question. Furthermore, comparability across different data sets is hopeless - different conventions and formats are almost always used.

It seems clear that in order for experimental software engineering to advance, these issues must be addressed - after all, an experimental science is based on collecting and analyzing data. However, without readily available, consistent and reliable data we will never be able to draw meaningful conclusions. Two types of data are of interest to researchers: product data and process data. Product data refers to the measurement of the product (typically the software itself) while process data refers to the measurement of the process (eg, time to develop, number of code errors, etc.). This paper focuses on the problem of collecting, maintaining and sharing product data. While not necessarily any more important than process data, the ability to realistically automate the collection of product data suggests this as the more practical avenue of effort.

2 Problems With the Current State of the Practice

Perhaps the most visible symptom of the problems with the current state of software product measurement is the inconsistency of conclusions. A case in point is Software Complexity Metrics. Since the early 1970's, countless experimental studies have been published, some in support of various Complexity Metrics, others showing various Complexity Metrics are virtually worthless. There are at least two foundational problems which contribute to this symptom.

2.1 Inability to Use Prior Data Sets

First and foremost, virtually every study published is based on a different data set. In some cases, this is desirable. For instance, if one is trying to validate a metric which has been previously proposed and evaluated using empirical data, bringing additional data to bear on the problem can add to our understanding of the metric. On the other hand, if a new metric, or some modification of an existing metric is proposed, one should be able to see how the new metric or modification fares when applied to prior data sets before we even start looking for new data sets. Unfortunately, such a scenario

is currently out of the question. With a few notable exceptions, such as the SEL and DACS collections, data is almost never shared among unrelated researchers. Thus, we will often see a new metric or modification that performs much better than an existing metric on a new data set, but will be unable to assess its performance on data from prior studies. Thus, we start from scratch each time a new metric is proposed. The net result is that we may have to reject years of accumulated data analysis on the basis of one or two new studies. Rather than "standing on the shoulders of giants" we end up stepping on each other's toes.

In fairness, of course, we must recognize that in the case of industrial product data, a researcher is fortunate indeed to simply be able to collect and retain the metrics. It is virtually unheard of for a researcher to be allowed to also keep the source code. Thus, even if researchers may be inclined to share their data, future analysis will be limited to whatever metrics happened to be collected during the original field work.

2.2 Inconsistent Counting Rules

Often a study purports to be a validation (or invalidation, as the case may be) of an existing metric, when in reality, due to subtle differences in counting rules it is actually a modification of the metric. Often such a situation is noticed when the results of studies appear strangely inconsistent. For instance, in a hypothetical example, one study may report on the superior performance of Software Science measures computed on a file-by-file basis in identifying error prone modules. Subsequently, a later study in which Software Science measures are computed on a function-by-function basis may report poor performance at exactly the same task.

When researchers go out of their way to replicate the counting rules of prior studies the descriptions of the counting rules used are usually incomplete. For instance, in the case of the C programming language, how are ifdef's counted? The code in the ifdefs? Are define'd constants that represent operators (eg, #define BEGIN {) treated as operators or operands? The only adequate description of the counting rules used is the source code of the analyzer that extracts the measures.

3 Well-Defined, Shareable Product Representations

No tool, proposal of data collection methodologies or other technical innovation can change the willingness of a researcher to share their data. This can only be addressed by the reward system within which most of us work. Proposing ways to encourage the sharing of data is outside the scope of this proposal. We will limit our discussion to those problems which have a "technical solution".

It is clear that any technical solution must address two points: (1) data must be available in a form that allows researchers in the future to explore different measures or counting rules than the ones originally used so prior data can be used to support new metrics, while at the same time ensuring that the original source code cannot be regenerated and (2) the data must support a way to completely and concisely articulate the