How Open is Open Source? Metrics for measuring software mutation

Charles Hathaway

February 18, 2015

Abstract

Open source software is toted as being "openly accessible" to many people, thus allowing greater innovation and complex new methodologies. However, given the complexity of software and how difficult it is to design and implement, the question of whether or not secondary communities adapt and modify the software needs to be addressed. This paper will first summarize previous works regarding the structure of open source communities, discuss how useful or not useful previous attempts to quantify the "open-ness" of projects have been, and finally propose a metric for measuring how open a project is. We will conclude with a proposal for a technique to test the proposed metric.

1 Introduction

2 Literature Review

2.1 A case study of the evolution of Jun: an object-oriented open-source 3D multimedia library

This paper discusses the development of an open source graphics library that focuses on ease of use over performance. Some notable things to consider; the project was written in Smalltalk (the same language as the original Scratch), it still exists, and it was primarily developed by a small team (rather than a community)

?? uses graph theory to map developers to projects, and create a "graph" which they discuss in the form of network theory. This is very similiar to what I had hoped to do with Github, and the graph is fascinating. They coin the term "linchpin developer" to talk about developers who tie together projects. Very interesting paper, but it needs better formatting...

3 Discussion and comparison of previous metrics

3.1 Cyclomatic complexity

[2]

3.2 Normalized Compression Distance

[1]

3.3 Effort measure

[5]

3.4 Data flow complexity

[3]

3.5 Complexity Measure Based on Program Slicing (CMBPS)

[4]

4 A new metric

5 Experiment proposals

6 Conclusion

7 References

References

[1] Rudi Cilibrasi and Paul MB Vitányi. "Clustering by compression". In: *Information Theory, IEEE Transactions on* 51.4 (2005), pp. 1523–1545.

[2] Thomas J. McCabe. "A Complexity Measure". In: *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING* SE-2.4 (1976), pp. 308–320.

[3] Enrique I Oviedo. "Control flow, data flow and program complexity". In: *Software engineering metrics I.* McGraw-Hill, Inc. 1993, pp. 52–65.

[4] Hongwei Tao and Yixiang Chen. "Complexity measure based on program slicing and its validation". In: *Wuhan University Journal of Natural Sciences* 19.6 (2014), pp. 512–518.

[5] Elaine J. Weyuker. “Evaluating Software Complexity Measures”. In: *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING,* 14.9 (1988), pp. 1357–1365.