Longitudinal Evaluation of Open-Source Software Maintainability

The most recent standardized model is ISO25010 (ISO/IEC 25010, 2011), in which maintainability is comprised of sub-characteristics Modularity, Reusability, Analyzability, Modifiability and Testability.

Maintainability Index

The Maintainability Index (MI) was introduced in1992 (Oman and Hagemeister, 1992)

MI=171−5.2∗ln(aveV)−0.23∗aveG−16.2∗ln(aveST AT), where

aveV- average Halstead volume. It reflects thecomputational load of the code in terms of operators and operands used.

aveG- average cyclomatic complexity. It reflectsthe number of possible execution paths.

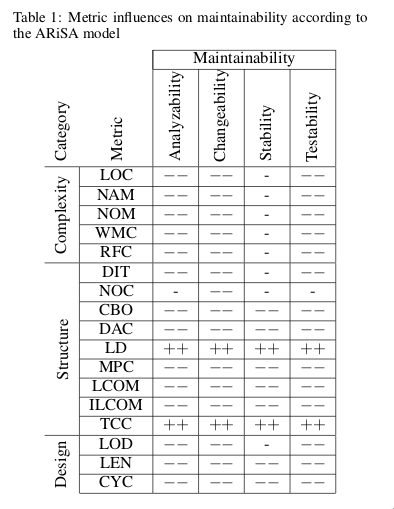
aveSTAT- average number of source code state-ments. Variants exist using the number of lines ofcode, however the consideration is (Virtual Ma-chinery, 2019) that the number of statements pro-vides a better reflection of source code size

Technical Debt Ratio

SonarQube is perhaps the most well known tool that calculates technical debt. It employs a configurable model where each issue discovered using static analysis is associated with an estimation of the time required to fix it. The technical debt represents the total time required to fix all discovered issues. The technical debt ratio puts the debt into context, by dividing the amount of time required to fix all issues by the estimated time to create the software system1.

AriSA Model for Maintainability

The ARiSA Compendium explores the relation between software quality, as expressed by the ISO9126 standard, and software metrics. The ISO9126 quality model is comprised of six characteristics and 27 sub-characteristics.



Metric values can be extracted using the Vizz Maintenance (ARISA Compendium, VizzMain-tenance, 2019) Eclipse plugin, which can extract values for the following class-level metrics: lines of code(LOC), number of attributes and methods (NAM),number of methods (NOM), weighted method count(WMC), response for class (RFC), depth of inheritance tree (DIT), number of children (NOC), coupling between objects (CBO), data abstraction coupling (DAC), locality of data (LD), message pass coupling (MPC), lack of cohesion in methods (LCOM)and its improved variant (ILCOM), tight class cohe-sion (TCC), lack of documentation (LOD), length ofnames (LEN) and number of classes in cycle (CYC).