Measure and Visualize Source Code Quality

Jeremy Ludwig & Devin Cline
ludwig; dcline @ stottlerhenke.com

INTRODUCTION
Creating and maintaining high-quality software is especially important for critical systems such as those made for NASA and the DoD, and for software product lines where long-lived, reusable modules are intended to be shared by multiple systems.

The goal of CBR Insight (CBRI) is to provide an objective and understandable measure of software quality that can help guide decisions during software acquisition, development, and sustainment.

DASHBOARD
Present an overall score along with scores for architecture, complexity, and clarity for a set of projects – the top three sources of avoidable technical debt.

Color-coded arrows describe change relative to a baseline to highlight the current direction of changes.

SCORING & BENCHMARKS
- Examines source code to calculate metrics that are highly related to software reliability, maintainability, and preventable technical debt [1-3] using a custom plug-in for SciTool’s Understand software [4].
- Develops realistic target ranges for these metrics based on successful, carefully selected, ‘peer’ projects selected from GitHub.
- Generates an aggregated score by comparing metric calculations to the target ranges.

PROJECT VIEW
Provides a description of the metrics and visualizes the measurements over time.

BACKGROUND AND RELATED WORK
There is an abundance of related work in software quality, technical debt, and automated code review that identifies specific source code metrics, describes how the measurements of these metrics are aggregated, and how the aggregations are used to assess characteristics of software quality and technical debt.

Summarizing this work is outside the scope of this abstract, see [5], [6] as a starting point. See [7] for a more detailed discussion of the metrics selected for use in CBRI.

REFERENCES