The Delta Maintainability Model: Measuring Maintainability of Fine-Grained Code Changes

di Biase, Marco; Rastogi, Ayushi; Bruntink, Magiel; van Deursen, Arie

DOI
10.5281/zenodo.2606632
10.1109/TechDebt.2019.00030

Publication date
2019

Document Version
Other version

Published in
Proceedings - 2019 IEEE/ACM International Conference on Technical Debt, TechDebt 2019

The Delta Maintainability Model: Measuring Maintainability of Fine-Grained Code Changes Technical Report

Marco di Biase*†, Ayushi Rastogi†, Magiel Bruntink*, Arie van Deursen†
*Software Improvement Group - Amsterdam, The Netherlands †Delft University of Technology - Delft, The Netherlands
Email: *[m.dibiase, m.bruntink]@sig.eu, †[a.rastogi, Arie.vanDeursen]@tudelft.nl

I. Introduction

This technical report contains definitions for the paper 'The Delta Maintainability Model: Measuring Maintainability of Fine-Grained Code Changes' submitted to the 2nd International Conference on Technical Debt (TechDebt 2019) - Montréal, Canada - May 26–27, 2019.

II. Defintions

Figure 1 provides an overview of the model and its underlying calculations.

The calculation of the DMM consists of two levels. The first level (Figure 1a) defines how a code change is mapped into several Risk Profile Deltas, which are an extension of the Risk Profile concept used in the SIG-MM. Then, the second level (Figure 1b) defines how the individual Risk Profile Deltas are combined to create a Delta Score at the commit level.

The input to DMM is a delta, or code change, consisting of the source code files that were involved in the change. More formally, we define a delta D as a set of n tuples $\langle f', f\rangle$, with f' and f the versions of a file prior to and after the change:

Definition 1 — *Delta.*

$$D = \{\langle f'_i, f_i\rangle | i \in \{1..n\}\}.$$

Risk profile deltas are defined with respect to the set *CP* of relevant code properties, of which we have five, and a set of four Risk Categories:

Definition 2 — *Code Properties, Risk Categories.*

$$\begin{aligned} \mathit{CP} &= \langle \textit{Unit Size, Unit Complexity, Unit Interfacing},\\ &\quad \textit{Duplication, Module Coupling}\rangle \\ \mathit{RC} &= \langle \textit{low, med, high, very-high}\rangle \end{aligned}$$

Next, we build up our definition of risk increases and decreases. We define a Risk Profile Delta between files f' and f, relative to a Code Property cp and Risk Category r:

Definition 3 — *Risk Profile Delta (RPD).*

$$\mathit{RPD}(f', f, cp, r) = \mathit{LOC}(f, cp, r) - \mathit{LOC}(f', cp, r)$$

In definition 3, *LOC* maps its arguments to the number of Lines Of Code in the Risk Category r of the code property cp for a specific file f. All Code Properties are mapped to Lines Of Code as specified by the SIG-MM [1] and all measurements are computed at file level.

Risk Profile Deltas are separated into increases and decreases:

Definition 4 — *Risk Profile Delta Increase (RPDI) and Decrease (RPDD).*

$$\begin{aligned} \mathit{RPDI}(f', f, cp, r) &= \mathit{max}(0, \mathit{RPD}(f', f, cp, r)) \\ \mathit{RPDD}(f', f, cp, r) &= |\, \mathit{min}(0, \mathit{RPD}(f', f, cp, r))\,| \end{aligned}$$

With this, we have the increases and decreases for each property at each Risk Category and for each file, as shown at the bottom of Figure 1a in step 4. We next aggregate these to the commit level, following Figure 1b.

Definition 5 — *Commit Risk Profile Delta Increase (CRPDI) and Decrease (CRPDD).*

$$\begin{aligned} \mathit{CRPDI}(cp, r) &= \sum_{\langle f', f\rangle \in D} \mathit{RPDI}(f', f, cp, r) \\ \mathit{CRPDD}(cp, r) &= \sum_{\langle f', f\rangle \in D} \mathit{RPDD}(f', f, cp, r) \end{aligned}$$

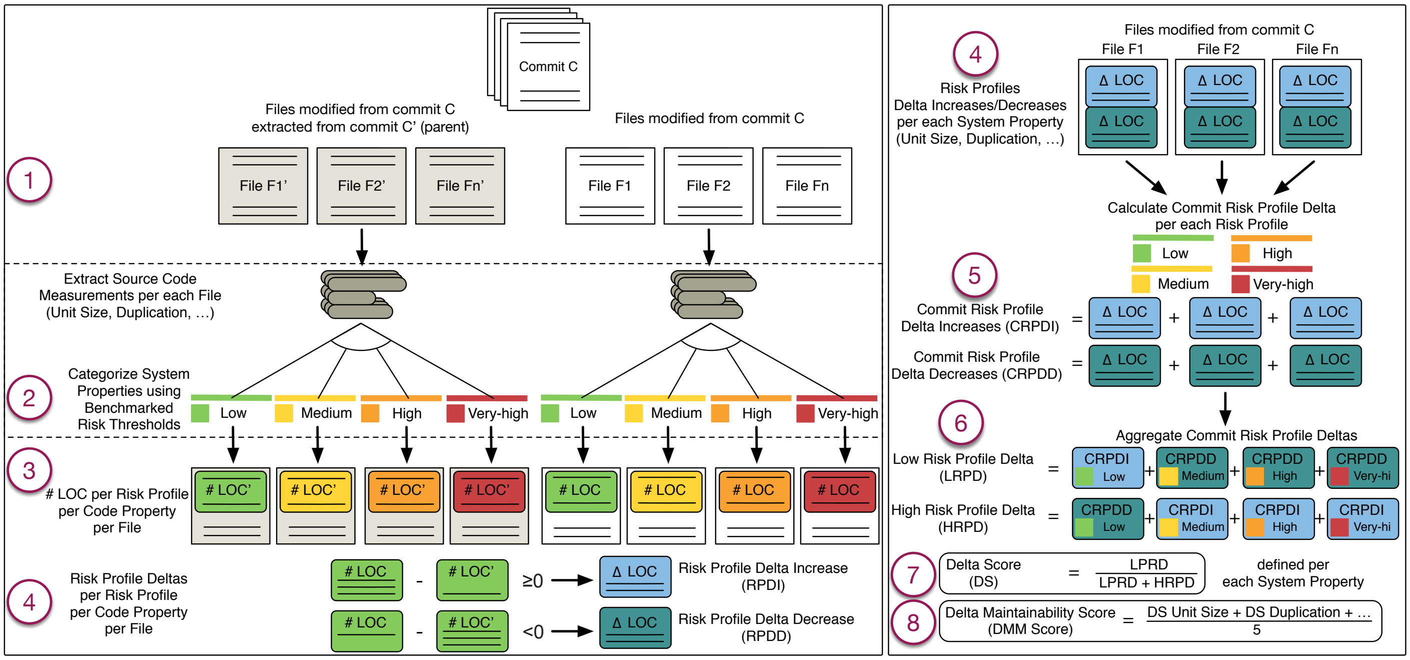
To detect which components of the definition 5 have high maintainability, we define Low Risk Profile Delta as the *increase* in low risk code summed to a *decrease* in medium/high/very-high risk code (both of which are good).

Definition 6 — *Low Risk Profile Delta (LRPD).*

$$\begin{aligned} \mathit{LRPD}(cp) = \mathit{CRPDI}(cp, \mathit{low}) + \\ \sum_{h \in \{\textit{medium, high, very-high}\}} \mathit{CRPDD}(cp, h) \end{aligned}$$

The goal of definition 6 is to sum Commit Risk Profile Deltas that are highly maintainable. In fact: 1) $CRPDI(cp, \text{low})$ adds code in the low risk category; therefore, it is highly maintainable; 2) $CRPDD(cp, \text{medium}) + CRPDD(cp, \text{high}) + CRPDD(cp, \text{very-high})$ removes code from the higher risk categories; therefore, it is highly maintainable.

Similarly, we have to define components of the definition 5 that have low maintainability. Therefore, we define High Risk Deltas as a decrease in low risk code, or an increase in code that is not low risk:



(A) Level 1: risk profile deltas.

(B) Level 2: delta scores.

Figure 1. Overview of the DMM.

Definition 7 — *High Risk Profile Delta (HRPD).*

$$HRPD(cp) = CRPDD(cp, low) + \sum_{h \in \{medium,\ high,\ very\text{-}high\}} CRPDI(cp, h)$$

The goal of definition 7 is dual to that of definition 6, *i.e.*, to sum Commit Risk Profile Deltas that are less maintainable. In fact: 1) $CRPDD(cp, \text{low})$ removes code in the low risk category; therefore, its maintainability is low; 2) $CRPDI(cp, \text{medium}) + CRPDI(cp, \text{high}) + CRPDI(cp, \text{very-high})$ adds code from the higher risk categories; therefore, they are less maintainable.

The Delta Score for a Code Property cp then is the proportion of low risk delta:

Definition 8 — *Delta Score (DS).*

$$DS(cp) = \frac{LRPD(cp)}{LRPD(cp) + HRPD(cp)}$$

The rationale behind definition 8 is that we compute the amount of Low Risk Profile Delta over the total amount of Risk Profile Delta measured. Given a Code Property, the resulting value represents the percentage of highly maintainable Delta Risk Profile over its total. Finally, the Delta Maintainability Model Score is the mean of all the single Code Properties in one value:

Definition 9 — *Delta Maintainability Model (DMM) Score.*

$$DMM\ Score = \frac{\sum_{cp \in CP} DS(cp)}{|CP|}$$

References

[1] T. L. Alves, J. P. Correia, and J. Visser. Benchmark-Based Aggregation of Metrics to Ratings. In *2011 Joint Conference of the 21st International Workshop on Software Measurement and the 6th International Conference on Software Process and Product Measurement*, pages 20–29, 2011.