Maintainability Index

Yegor Bugayenko

Lecture #5 out of 24

80 minutes

The slidedeck was presented by the author in this YouTube Video

"The total <u>cost of maintaining</u> a widely used program is typically 40 percent or more of the cost of developing it"

— Fred Brooks, *The Mythical Man-Month: Essays on Software Engineering*, 1982

"Shipping first time code is like going into <u>debt</u>. A little debt speeds development so long as it is paid back promptly with a rewrite. The danger occurs when the debt is not repaid. Every minute spent on not-quite-right code counts as interest on that debt."

— Ward Cunningham, *Experience Report — The WyCash portfolio management system*, OOPSLA, 1992
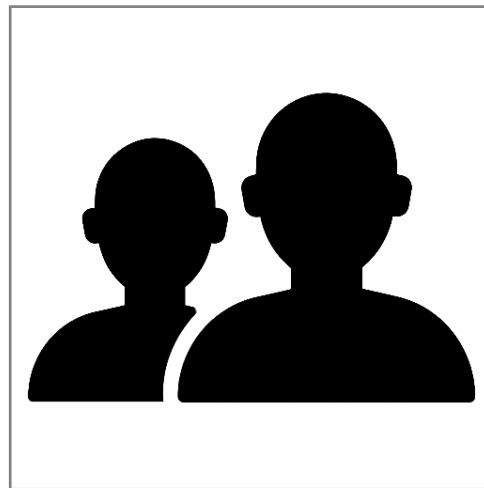
"Before developers can claim that they are building maintainable systems, there must be some way to measure maintainability"

— Don Coleman and Dan Ash (Hewlett Packard), Bruce Lowther (Micron Semiconductor), Paul Oman (University of Idaho), *Using Metrics to Evaluate Software System Maintainability*, 1994
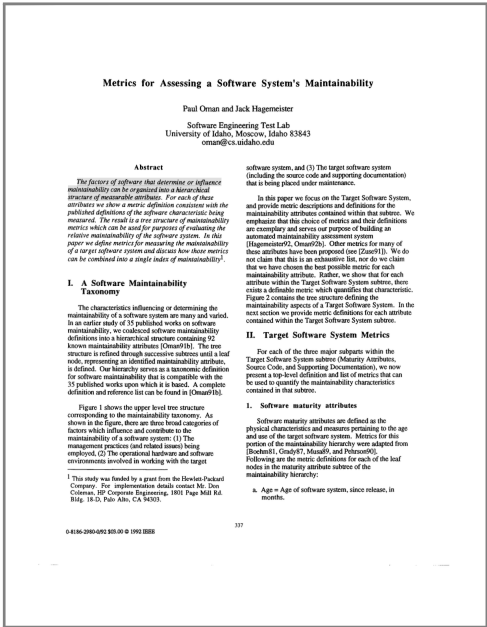
"The factors of software that determine or influence maintainability can be organized into a hierarchical structure of measurable attribute. Our hierarchy serves as a taxonomic definition for software maintainability that is compatible with the 35 published works upon which it is based."

— Paul Oman and Jack Hagemeister. Metrics for Assessing a Software System's Maintainability. In *Proceedings of the International Conference on Software Maintenance*, pages 337–338. IEEE Computer Society, 1992

Software Maintainability Taxonomy
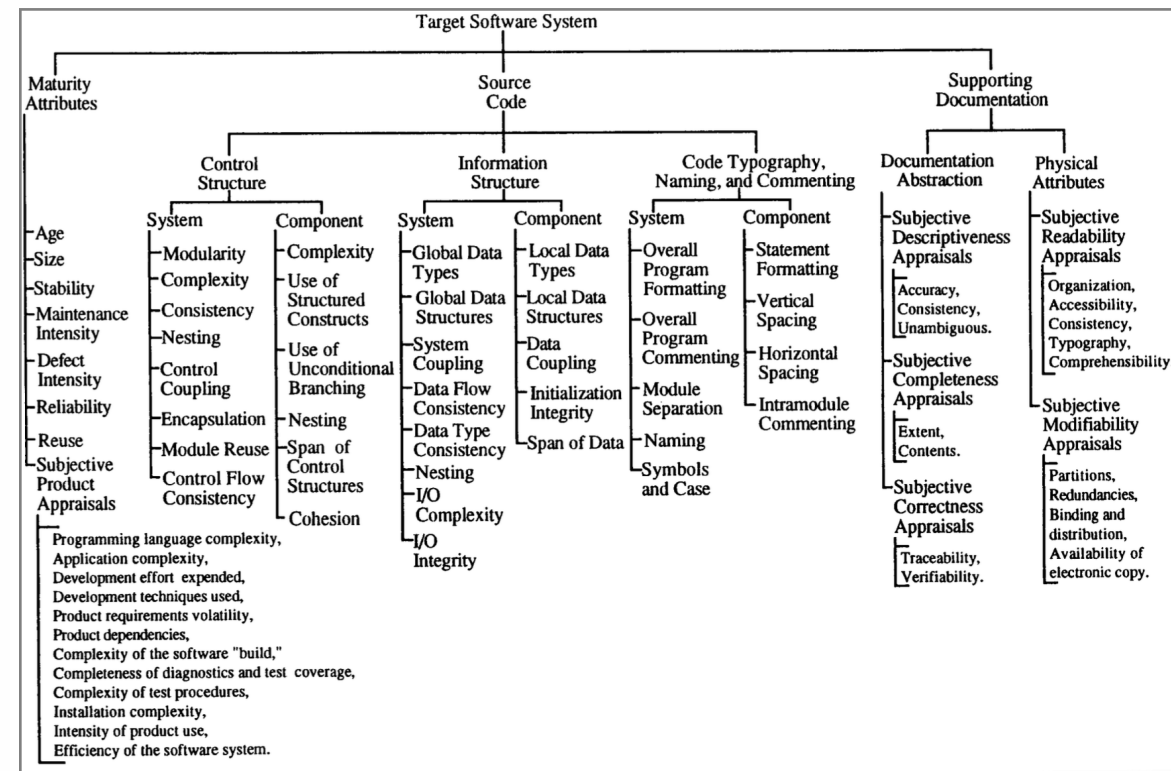
Source: Paul Oman and Jack Hagemeister. Metrics for Assessing a Software System's Maintainability. In *Proceedings of the International Conference on Software Maintenance*, pages 337–338. IEEE Computer Society, 1992

Maintainability Formula

To quantify the maintainability of a tree we can then use the following formula:

$$\prod_{i=1}^{m} W_{D_i} \left(\frac{\sum_{j=1}^{n} W_{A_j} M_{A_j}}{n} \right)_i$$

where

W_{D_i} = Weight of influence of maintainability Dimension D_i

W_{A_j} = Weight of influence of maintainability Attribute A_j

M_{A_j} = Metric or measure of maintainability Attribute A_j

"This formula represents the product of the weighted dimensions, where each dimension is measured as the average deviation from a known value of 'goodness' for that maintainability attribute."

Source: Paul Oman and Jack Hagemeister. Metrics for Assessing a Software System's Maintainability. In *Proceedings of the International Conference on Software Maintenance*, pages 337–338. IEEE Computer Society, 1992
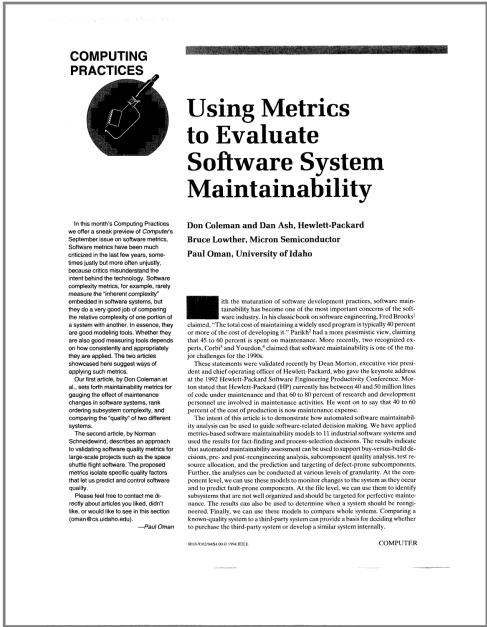
"A software maintainability model is only useful if it can provide developers and maintainers in an industrial setting with more information about the system"

— Don Coleman, Dan Ash, Bruce Lowther, Paul Oman, *Using Metrics to Evaluate Software System Maintainability*, 1994

First Approximation

$$\text{Maintainability} = 171$$
$$- 3.42 \times \ln(aveE)$$
$$- 0.23 \times aveV(g')$$
$$- 16.2 \times \ln(aveLOC) + aveCM$$

where $aveE$, $aveV(g')$, $aveLOC$, and $aveCM$ are the average effort, extended V(G), average lines of code, and number of comments per submodule (function or procedure) in the software system.

"Approximately 50 regression models were constructed in an attempt to identify simple models that could be calculated from existing tools and still be generic enough to apply to a wide range of software systems. The regression model that seemed most applicable was a four-metric polynomial based on 1) Halstead's effort, 2) extended cyclomatic complexity, 3) lines of code, and 4) number of comments."

The Formula of Maintainability Index

$$\text{Maintainability} = 171$$
$$-5.2 \times \ln(aveVol)$$
$$-0.23 \times ave\ V(g')$$
$$-16.2 \times \ln(aveLOC)$$
$$+(50 \times \sin(\sqrt{2.46 \times perCM}))$$

aveVol — average Halstead Volume in a module

ave V(g') — average total cyclomatic complexity in a module

aveLOC — average lines of code in a module

perCM — average percent of comments in a module

Maintainability Index by Visual Studio

$$MI = \max \left[0, 100 \frac{171 - 5.2 \ln V - 0.23G - 16.2 \ln L}{171}\right]$$

Source: Introduction to Code Metrics, by Radon

🟩	MI >= 20	High Maintainabillity
⚠️	10 <= MI < 20	Moderate Maintainability
🔴	MI < 10	Low Maintainability

Source: Think Twice Before Using the "Maintainability Index", by Arie van Deursen

"We decided to be conservative with the thresholds. The desire was that if the index showed red then we would be saying with a high degree of confidence that there was an issue with the code." — Code metrics — Maintainability index range and meaning by Microsoft, 2011.

"We are convinced that Maintainability Index is nonsense. We think that it is not sensible to reduce the maintainability of a whole software system to one single indicator."

— Rainer Niedermayr, *Why we don't use the Software Maintainability Index*, 2016, link

"The Maintainability Index does not provide information about the impact on development activities. A value of 57 does not express which maintainability aspects are affected by a bad value." — Rainer Niedermayr

"If you are a researcher, think twice before using the maintainability index in your experiments. Make sure you study and fully understand the original papers published about it."

— Arie van Deursen, *Think Twice Before Using the "Maintainability Index"*, 2014, link

"Tool smiths and vendors used the exact same formula and coefficients as the 1994 experiments, without any recalibration." — Arie van Deursen

"If we're going to use the Maintainability Index we should use it to measure <u>relative</u> maintainability within our project rather than use it as an <u>absolute</u> metric."

— Tim Gilboy, *Maintainability Index - What is it and where does it fall short?*, 2022, <u>link</u>

"Extending the length can significantly decrease Maintainability Index, even if all of the changes cause the code to be clearer and more understandable." — Tim Gilboy
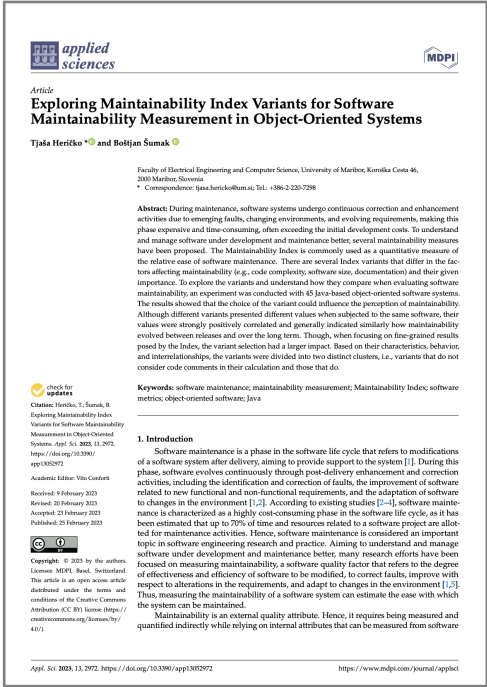
"When comparing maintainability measurements from several Index variants, the perception of maintainability could be impacted by the choice of the Index variant used."

— Tjaša Heričko et al., *Exploring Maintainability Index Variants for Software Maintainability Measurement in Object-Oriented Systems*, Applied Sciences, 2023

Maintainability Index is supported by a few tools:

- Visual Studio for C++ and others

- SonarQube for Java

- Testwell for Java and C++

- Radon for Python

- jscomplexity for JavaScript

- maintidx for Go
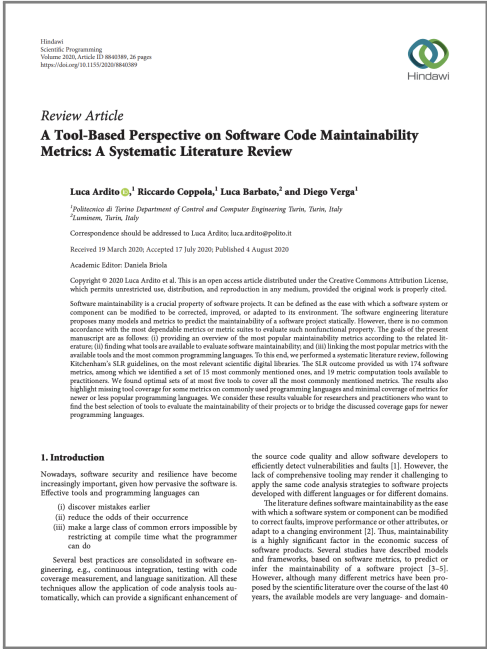
"The SLR outcome provided us with 174 software metrics, among which we identified a set of 15 most commonly mentioned ones, and 19 metric computation tools available to practitioners."

— *A Tool-Based Perspective on Software Code Maintainability Metrics: A Systematic Literature Review*, Luca Ardito et al., Scientific Programming, 2020

Milestones of Your Research

1. Research Question(s)

2. Research Method

3. Experiments

4. Related Work

5. Results

6. Limitations

7. Discussion

8. Conclusion

9. Introduction

10. Abstract

11. Title

References

Paul Oman and Jack Hagemeister. Metrics for Assessing a Software System's Maintainability. In *Proceedings of the International Conference on Software Maintenance*, pages 337–338. IEEE Computer Society, 1992.