Cyclomatic Complexity

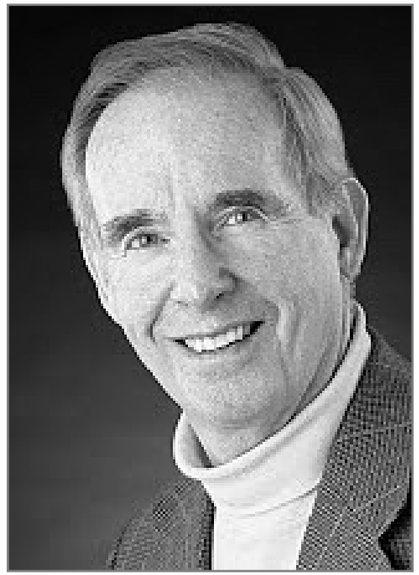
Yegor Bugayenko

Lecture #2 out of 24
80 minutes

The slidedeck was presented by the author in this YouTube Video

All visual and text materials presented in this slidedeck are either originally made by the author or taken from public Internet sources, such as web sites. Copyright belongs to their respected authors.

1. Some programmers mistakenly feel proud of higher complexity of their code.



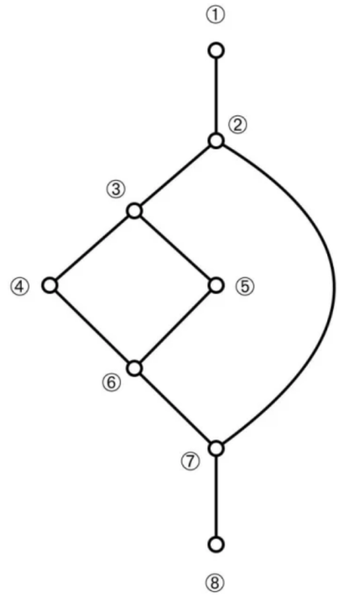
Thomas J. McCabe

"Cyclomatic Complexity (CC) is a count of the number of decisions in the source code. The higher the count, the more complex the code. The formula is simple: $C = E - N + 2$."

— Thomas J. McCabe. A Complexity Measure. *IEEE Transactions on Software Engineering*, (4):308–320, 1976. doi:10.1109/tse.1976.233837

What is the complexity of this program?

```
① start
② if (X) then
③     if (Y) then
④         perform A
          perform B
      else
⑤         perform C
          perform D
⑥     endif
⑦ endif
⑧ end
```



I found this picture here.

2. In his presentation "*Software Quality Metrics to Identify Risk*", Tom McCabe introduces the following categorisation to interpret cyclomatic complexity: 1–10 little risk, 11–20 moderate risk, 21–50 high risk, 50+ very high risk.

Joanne E. Hale

"The models developed are found to successfully predict roughly 90% of CC's variance by LOC alone. This suggest not only that the linear relationship between LOC and CC is stable, but the aspects of code complexity that CC measures, such as the size of the test case space, grow linearly with source code size across languages and programming paradigms."

— Jay Graylin, Joanne E. Hale, Randy K. Smith, Hale David, Nicholas A. Kraft, and Ward Charles. Cyclomatic Complexity and Lines of Code: Empirical Evidence of a Stable Linear Relationship. *Journal of Software Engineering and Applications*, 2(03):137, 2009. doi:10.4236/jsea.2009.23020

Adnan Muslija

"There is a low to moderate correlation between the effort needed to test a program and its complexity."

— Adnan Muslija and Eduard P. Enoiu. On the Correlation between Testing Effort and Software Complexity Metrics, 2018

Gregory Seront

"From the results of the experiments we conducted, we observed no significant correlation between the depth of inheritance of a class and its weighted method complexity."

— Grégory Seront, Miguel Lopez, Valérie Paulus, and Naji Habra. On the Relationship Between Cyclomatic Complexity and the Degree of Object Orientation. In *Proceedings of 9th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE)*, pages 109–117, 2005

Abd Jader

"As the complexity of the software increases, the probability to introduce new errors also increases."

— Abd Jader, Marwa Najm, and Riyadh Zaghlool Mahmood. Calculating McCabe's Cyclomatic Complexity Metric and Its Effecct on the Quality Aspects of Software. *International Journal of Innovative Research and Creative Technology*, (5), 2018

Meine van der Meulen

> “There is a very strong correlation between Lines of Code and Halstead Volume and there is an even stronger correlation between Lines of Code and McCabe’s Cyclomatic Complexity.”
>
> — Meine J. P. van der Meulen and Miguel A. Revilla. Correlations between Internal Software Metrics and Software Dependability in a Large Population of Small C/C++ Programs. In *Proceedings of the 18th International Symposium on Software Reliability (ISSRE’07)*, pages 203–208. IEEE, 2007

Yonghee Shin

"The results of our study show weak evidence that software complexity is the enemy of software security for the nine complexity metrics we collected. However, vulnerable code seems to be more complex than faulty code."

— Yonghee Shin and Laurie Williams. Is Complexity Really the Enemy of Software Security? In *Proceedings of the 4th ACM Workshop on Quality of Protection*, pages 47–50, 2008. doi:10.1145/1456362.1456372

Yonghee Shin

"The results of our study show weak evidence that software complexity is the enemy of software security for the nine complexity metrics we collected. However, vulnerable code seems to be more complex than faulty code."

— Yonghee Shin and Laurie Williams. Is Complexity Really the Enemy of Software Security? In *Proceedings of the 4th ACM Workshop on Quality of Protection*, pages 47–50, 2008. doi:10.1145/1456362.1456372

Abram Hindle

> “Our results strongly suggest that measuring indentation is a cheap and accurate proxy for code complexity of revisions.”

— Abram Hindle, Michael W. Godfrey, and Richard C. Holt. Reading Beside the Lines: Indentation as a Proxy for Complexity Metrics. In *Proceedings of the 16th International Conference on Program Comprehension*, pages 133–142. IEEE, 2008. doi:10.1109/ICPC.2008.13

Md Abdullah Al Mamun

"We also found that complexity and documentation domains are more correlated with size domain than themselves"

— Md Abdullah Al Mamun, Christian Berger, and Jörgen Hansson. Correlations of Software Code Metrics: An Empirical Study. In *Proceedings of the 27th International Workshop on Software Measurement and the 12th International Conference on Software Process and Product Measurement*, pages 255–266, 2017. doi:10.1145/3143434.3143445

3. What is a complexity of a class? How about a module?

> 4. Feature creep is one of the most common sources of cost and schedule overruns; it can even kill products and projects — Wikipedia.

5. Tom McCabe suggested to prohibit functions where CC is larger than ten. Modern static analyzers may help you control this automatically.

Geoffrey K. Gill

"The complexity density ratio is demonstrated to be a useful predictor of software maintenance productivity on a small pilot sample of actual maintenance project."

— Geoffrey K. Gill and Chris F. Kemerer. Cyclomatic Complexity Density and Software Maintenance Productivity. *IEEE Transactions on Software Engineering*, 17(12):1284–1288, 1991. doi:10.1109/32.106988

Read this:

The Better Architect You Are, The Simpler Your Diagrams (2015)

Are You a Hacker or a Designer? (2014)

References

Geoffrey K. Gill and Chris F. Kemerer. Cyclomatic Complexity Density and Software Maintenance Productivity. *IEEE Transactions on Software Engineering*, 17(12):1284–1288, 1991. doi:10.1109/32.106988.

Jay Graylin, Joanne E. Hale, Randy K. Smith, Hale David, Nicholas A. Kraft, and Ward Charles. Cyclomatic Complexity and Lines of Code: Empirical Evidence of a Stable Linear Relationship. *Journal of Software Engineering and Applications*, 2 (03):137, 2009. doi:10.4236/jsea.2009.23020.

Abram Hindle, Michael W. Godfrey, and Richard C. Holt. Reading Beside the Lines: Indentation as a Proxy for Complexity Metrics. In *Proceedings of the 16th International Conference on Program Comprehension*, pages 133–142. IEEE, 2008. doi:10.1109/ICPC.2008.13.

Meine J. P. van der Meulen and Miguel A. Revilla. Correlations between Internal Software Metrics and Software Dependability in a Large Population of Small C/C++ Programs. In *Proceedings of the 18th International Symposium on Software Reliability (ISSRE'07)*, pages 203–208. IEEE, 2007.

Abd Jader, Marwa Najm, and Riyadh Zaghlool Mahmood. Calculating McCabe's Cyclomatic Complexity Metric and Its Effecct on the Quality Aspects of Software. *International Journal of Innovative Research and Creative Technology*, (5), 2018.

Md Abdullah Al Mamun, Christian Berger, and Jörgen Hansson. Correlations of Software Code Metrics: An Empirical Study. In *Proceedings of the 27th International Workshop on Software Measurement and the 12th International Conference on Software Process and Product Measurement*, pages 255–266, 2017. doi:10.1145/3143434.3143445.

Thomas J. McCabe. A Complexity Measure. *IEEE Transactions on Software Engineering*, (4):308–320, 1976. doi:10.1109/tse.1976.233837.

Adnan Muslija and Eduard P. Enoiu. On the

Correlation between Testing Effort and Software Complexity Metrics, 2018.

Grégory Seront, Miguel Lopez, Valérie Paulus, and Naji Habra. On the Relationship Between Cyclomatic Complexity and the Degree of Object Orientation. In *Proceedings of 9th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE)*, pages 109–117, 2005.

Yonghee Shin and Laurie Williams. Is Complexity Really the Enemy of Software Security? In *Proceedings of the 4th ACM Workshop on Quality of Protection*, pages 47–50, 2008. doi:10.1145/1456362.1456372.