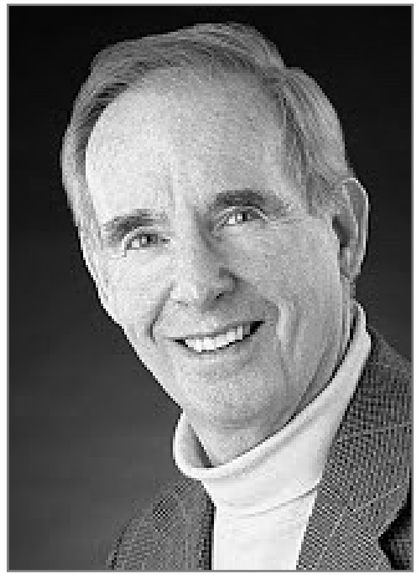
Cyclomatic Complexity

Yegor Bugayenko

Lecture #2 out of 24
80 minutes

The slidedeck was presented by the author in this YouTube Video

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



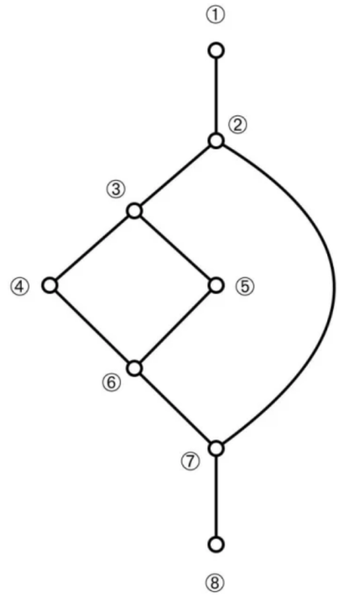
Thomas J. McCabe

"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.

3. Graylin JAY et al., *Cyclomatic Complexity and Lines of Code: Empirical Evidence of a Stable Linear Relationship*, Journal of Software Engineering & Applications, Vol. 2, 2009, pp. 137–143

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

> 5. Feature creep is one of the most common sources of cost and schedule overruns; it can even kill products and projects — Wikipedia.

6. 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.

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