

Coupling & Cohesion

and Other Metrics

YEGOR BUGAYENKO

Lecture #15 out of 16

90 minutes

All visual and text materials presented in this slidedeck are either originally made by the author or taken from public Internet sources, such as website.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

Size and Complexity Metrics

Coupling and Cohesion

Productivity and Its Metrics

Books, Venues, Call-to-Action

Chapter #1:

Size and Complexity Metrics

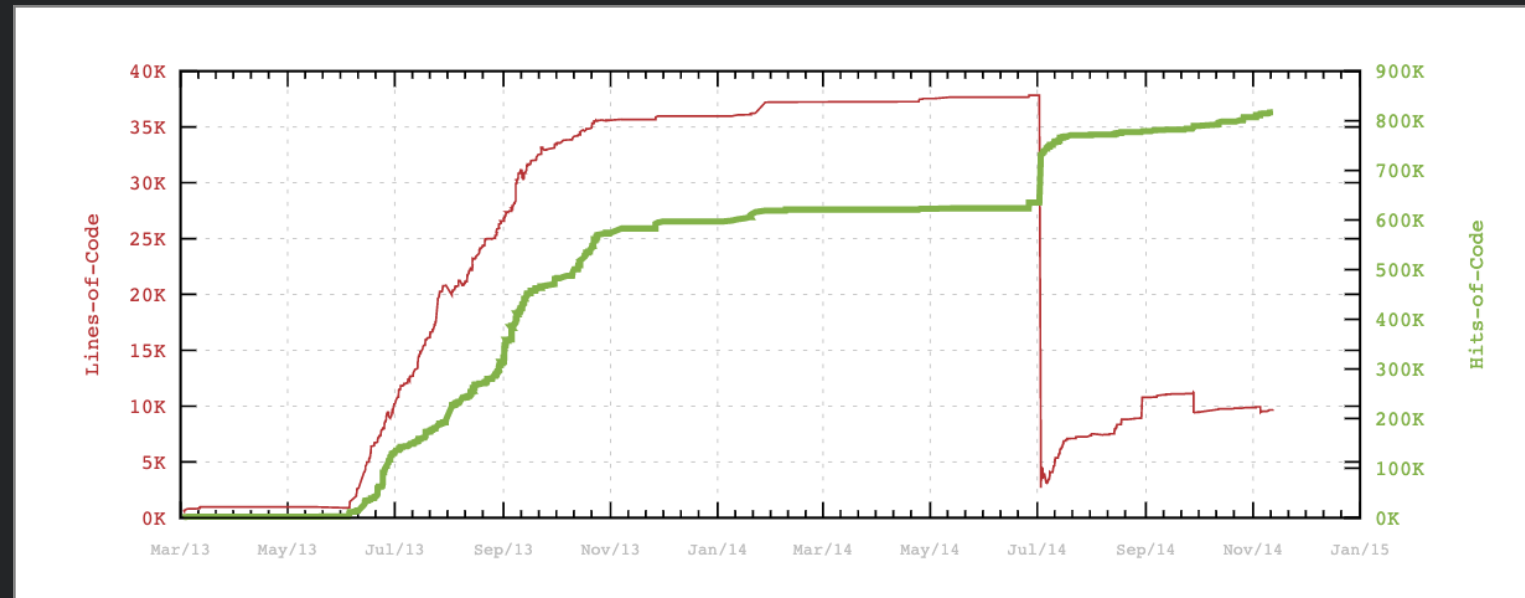
Software Lines of Code (SLOC)

```
/code/takes$ cloc .
  519 text files.
  517 unique files.
   16 files ignored.

github.com/AlDanial/cloc v 1.74  T=2.11 s (238.3 files/s, 25238.0 lines/s)
-----
Language                      files      blank      comment      code
-----
Java                          467        3289        22432       24646
Markdown                       9         236          0        1067
Maven                          2           8          58         678
XSLT                          14          56         266         219
YAML                           5           4           4         155
XML                            3          12          57          74
HTML                           2           8          38           57
Velocity Template Language     2           0           0           5
-----
SUM:                          504        3613        22855       26901
-----
```

<https://github.com/AlDanial/cloc>

Hits Of Code (HoC) or Code Churn



<https://www.yegor256.com/2014/11/14/hits-of-code.html>

<https://hitsofcode.com/>

J.C. Munson et al., *Code churn: a measure for estimating the impact of code change*, International Conference on Software Maintenance (ICSM), 1998

McCabe Cyclomatic Complexity (CC)

Mathematically, the cyclomatic complexity of a [structured program](#)^[a] is defined with reference to the [control-flow graph](#) of the program, a [directed graph](#) containing the [basic blocks](#) of the program, with an edge between two basic blocks if control may pass from the first to the second. The complexity **M** is then defined as^[2]

$$M = E - N + 2P,$$

where

E = the number of edges of the graph.

N = the number of nodes of the graph.

P = the number of [connected components](#).

Introduced by by Thomas J. McCabe, Sr. in 1976

Cognitive Complexity (CoC)

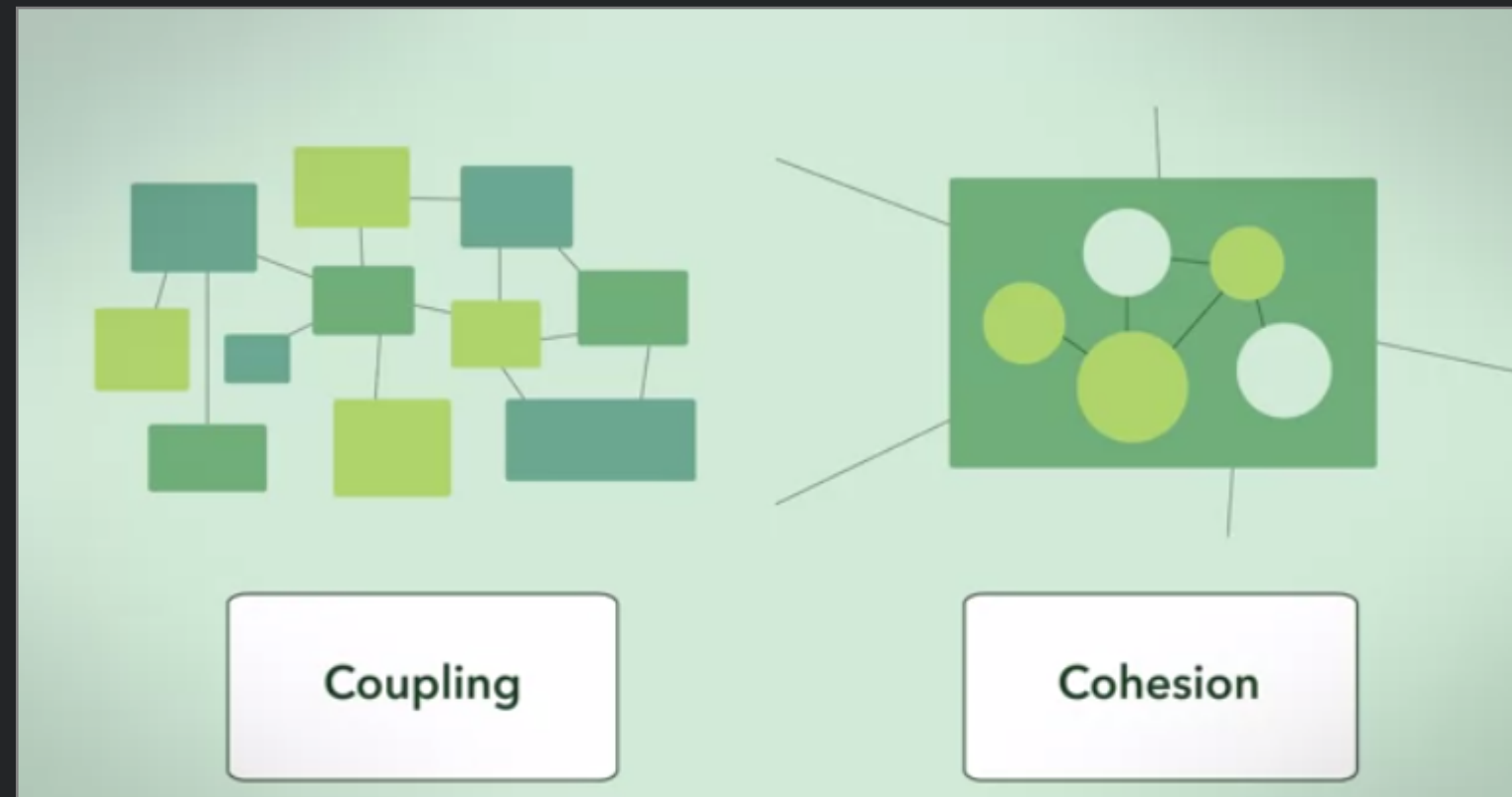
```
void myMethod () {  
    try {  
        if (condition1) {                // +1  
            for (int i = 0; i < 10; i++) { // +2 (nesting=1)  
                while (condition2) { ... } // +3 (nesting=2)  
            }  
        }  
    } catch (Exception1 | Exception2 e) { // +1  
        if (condition2) { ... }           // +2 (nesting=1)  
    }  
}                                           // Cognitive Complexity 9
```

by G. Ann Campbell, SonarSource, 2011

Chapter #2:

Coupling and Cohesion

Low Coupling and High Cohesion



Invented by Larry Constantine in the late 1960s as part of a structured design.

Lack of Cohesion of Methods (LCOM)

The Lack of Cohesion of Methods (**LCOM**) is a correlation between the methods and the local instance variables of a class (we use the version suggested by Henderson-Sellers et al. [19], also known as LCOM5). Let m be the number of methods, a be the number of attributes and μ_j be the amount of methods, which use attribute j , then,

$$LCOM = \frac{1}{1 - m} \left(\frac{1}{a} \sum_{j=1}^a \mu_j \right) - m.$$

Method-Method through Attributes Cohesion (MMAC)

The Method-Method through Attributes Cohesion (**MMAC**) metric, introduced by Dallal and Briand [13], is the average cohesion of all pairs of methods. Let k be the number of methods, l be the number of distinct parameter types, and x_i be the number of methods that use type i , then,

$$MMAC = \frac{1}{lk(k-1)} \sum_{i=1}^l x_i(x_i - 1).$$

Normalized Hamming Distance (NHD)

The Normalized Hamming Distance (**NHD**) class cohesion metric, introduced by Counsell et al. [11], measures the similarity in all methods of a class in terms of the types of their arguments. Let l be the number of distinct parameter types, k be the number of methods, and c_j be the number of methods that have a parameter of type j , then,

$$NHD = 1 - \frac{2}{lk(k-1)} \sum_{j=1}^l c_j(k - c_j).$$

Sensitive Class Cohesion Metric (SCOM)

The Sensitive Class Cohesion Metric (**SCOM**), introduced by Fernández and Peña [14], is a ratio of the summation of connection intensities $C_{i,j}$ of all pairs (i, j) of m methods to the total number of pairs of methods. Connection intensity must be given more weight $\alpha_{i,j}$ when such a pair involves more attributes:

$$SCOM = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^m C_{i,j} \times \alpha_{i,j}$$

Chapter #3:

Productivity and Its Metrics

Some Productivity Metrics

Features Delivered

Pull Requests Merged

Bugs Fixed

Bugs Reported

Releases Published

Uptime

Cost of Pull Request

Mentee Results



<https://www.yegor256.com/shift-m/2020/44.html>



Soft Skills

Drawing

Writing

Reporting

Branching

Asking

Charging

Relaxing



<https://www.yegor256.com/2018/08/29/soft-skills.html> →

Blame the Code, Not Yourself

<https://www.yegor256.com/2018/04/17/how-to-be-lazy.html>

<https://www.yegor256.com/2015/02/16/it-is-not-a-school.html>

Size C&C Productivity B.V.C.

18/25

[Metrics Soft NoBlame NoQuality NoMeetings NoBoss]

Aim for Speed, Not for Quality

<https://www.yegor256.com/2018/03/06/speed-vs-quality.html>

Avoid Meetings

<https://www.yegor256.com/2015/07/13/meetings-are-legalized-robbery.html>

<https://www.yegor256.com/2015/01/08/morning-standup-meetings.html>

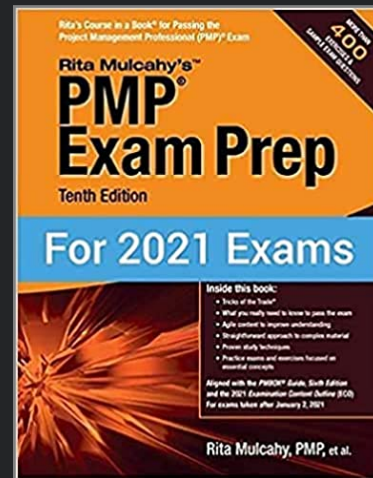
Work for Product, Not for Boss

<https://www.yegor256.com/2015/01/26/happy-boss-false-objective.html>

<https://www.yegor256.com/2015/02/23/haircut.html>

Chapter #4:

Books, Venues, Call-to-Action



“PMP Exam Prep: Tenth Edition” by
RITA MULCAHY



“Code Ahead” by YEGOR BUGAYENKO

Where to go:

International Conference on Software Metrics in Software Engineering
(ICSMSE)

Call to Action:

Configure automated collection of cohesion and other metrics in your project, and publish the numbers on each build.

Still unresolved issues:

- How to measure code readability?
- How to connect management and software metrics?
- How to balance different metrics?
- How to predict the future using metrics?