TCC and LCC

YEGOR BUGAYENKO

Lecture #8 out of 24 80 minutes

The slidedeck was presented by the author in this YouTube Video

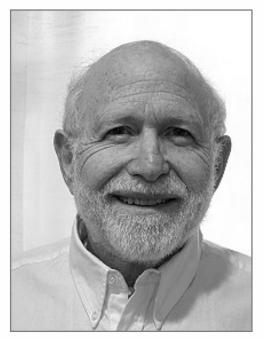
All visual and text materials presented in this slidedeck are either originally made by the author or taken from public Internet sources, such as web sites. Copyright belongs to their respected authors.



EDWARD YOURDON AND LARRY CONSTANTINE

"Module cohesion may be conceptualized as the <u>cement</u> that holds the processing elements of a module together. In a sense, a high degree of module cohesion is an indication of <u>close</u> approximation of inherent problem structure."

— Edward Yourdon and Larry Constantine. *Structured Design: Fundamentals of a Discipline of Computer Program and Systems Design.* Prentice Hall, 1979. doi:10.5555/578522



James M. Bieman

"We define two measures of class cohesion based on the <u>direct</u> and <u>indirect</u> connections of method pairs: TCC and LCC."

— James M. Bieman and Byung-Kyoo Kang. Cohesion and Reuse in an Object-Oriented System. *SIGSOFT Software Engineering Notes*, 20(SI):259–262, 1995. doi:10.1145/223427.211856

Tight and Loose Class Cohesion (TCC+LCC)

```
class Rectangle
int x, y, w, h;
int area()
return w * h;
int move(int dx, dy)
x += dx; y += dy;
int resize(int dx, dy)
w += dx; h += dy;
bool fit()
return w < 100
&& x < 100;
```

```
Max possible connections (NP):
```

$$N \times (N-1)/2 = 4 \times 3/2 = 6$$

Directly connected (NDC = 4):

area+fit, area+resize, move+fit,
resize+fit

Indirectly connected (NIC = 2):

area+move, move+resize

$$TCC = NDC/NP = 4/6 = 0.66$$

 $LCC = (NDC+NIC)/NP = 6/6 = 1.00$

Collection and Reuse in an Object-Oriented System

Lame No. Bimma and Disperts Oriented System

Lame No. Bimma and Disperts Oriented System

In Collection and Reuse in an Object-Oriented System

**In Collection and Reuse in an Objec

Object-Oriented System. SIGSOFT Software Engineering Notes, 20(SI):259–262,

TCC and LCC @yegor256

1995. doi:10.1145/223427.211856



STEVE McConnell

"Cohesion refers to how closely all the routines in a class or all the code in a routine support a central purpose—how focused the class is. The ideas of abstraction and cohesion are closely related—a class interface that presents a good abstraction usually has strong cohesion."

— Steve McConnell. *Code Complete*. Pearson Education, 2004. doi:10.5555/1096143

Abstraction



• Color: red

• Weight: 120g

• Price: \$0.99



```
var file = {
  path: '/tmp/data.txt',
  read: function() { ... },
  write: function(txt) { ... }
}
```

The slide is taken from the "Pain of OOP" (2023) course.

Inheritance vs. Cohesion

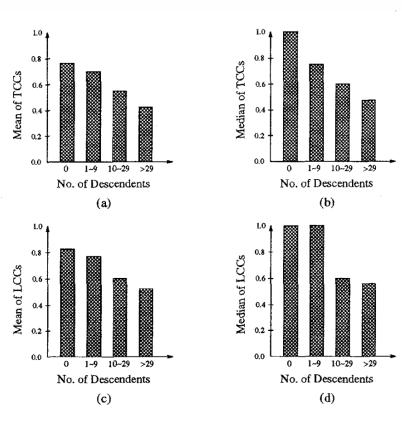


Figure 3: Number of descendents and Class Cohesion

"Our results show that the classes that are heavily reused via inheritance exhibit lower cohesion. We expected to find that the most reused classes would be the most cohesive ones." — James M. Bieman and Byung-Kyoo Kang

Inheritance is Code Reuse

```
class Manuscript {
  protected String body;
  void print(Console console) {
    console.println(this.body);
}

class Article
  extends Manuscript {
  void submit(Conference cnf) {
    cnf.send(this.body);
  }
}
```

"The Article copies method print() and attribute body from the Manuscript, as if it's not a living organism, but rather a dead one from which we inherit its parts."

"Implementation inheritance was created as a mechanism for code reuse. It doesn't fit into OOP at all."

Source: Inheritance Is a Procedural Technique for Code Reuse (2016)

Composition over Inheritance

```
class Manuscript
                                   | class Manuscript
                                      protected String body;
   protected String body;
   void print(Console console)
                                     void print(Console console)
     console.println(this.body);
                                        console.println(this.body);
                                   5
6 class Article
                                   6 class Article
   extends Manuscript
                                      Manuscript manuscript;
   void submit(Conference cnf)
                                      Article(Manuscript m)
     cnf.send(this.body);
                                        this.manuscript = m;
                                      void submit(Conference cnf)
                                        cnf.send(this.body);
                                   11
```

Wikipedia: https://en.wikipedia.org/wiki/Composition_over_inheritance

TCC+LCC can be calculated by a few tools:

- jPeek for Java
- C++ don't know
- Python don't know
- JavaScript don't know
- C# don't know

Read this:

Inheritance Is a Procedural Technique for Code Reuse (2016)

References

James M. Bieman and Byung-Kyoo Kang. Cohesion and Reuse in an Object-Oriented System. SIGSOFT Software Engineering Notes, 20(SI): 259–262, 1995. doi:10.1145/223427.211856. Steve McConnell. *Code Complete*. Pearson Education, 2004. doi:10.5555/1096143.

Edward Yourdon and Larry Constantine. *Structured Design: Fundamentals of a Discipline of Computer Program and Systems Design.* Prentice Hall, 1979. doi:10.5555/578522.