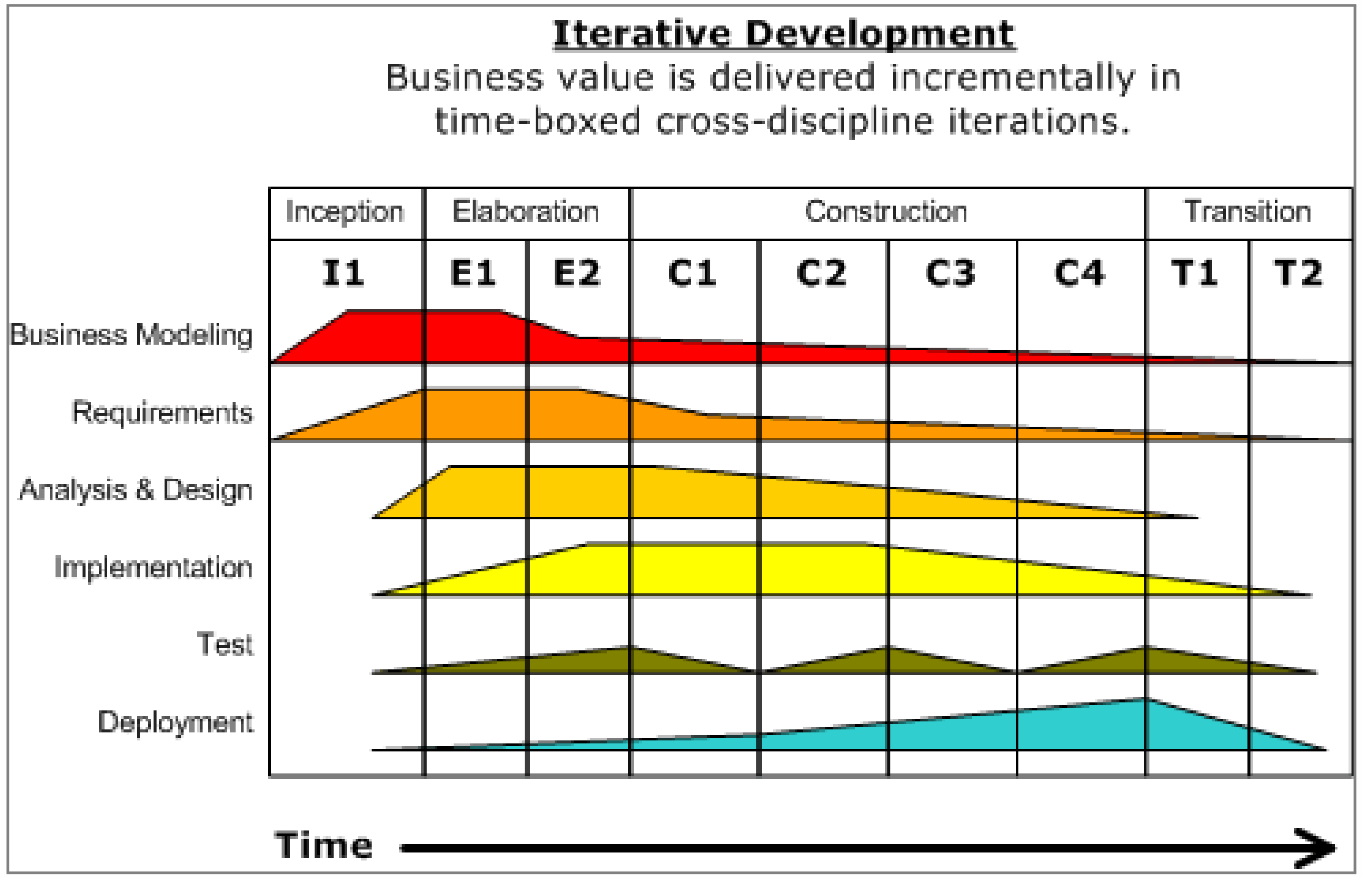
CAMC and NHD

Yegor Bugayenko

Lecture #9 out of 24
80 minutes

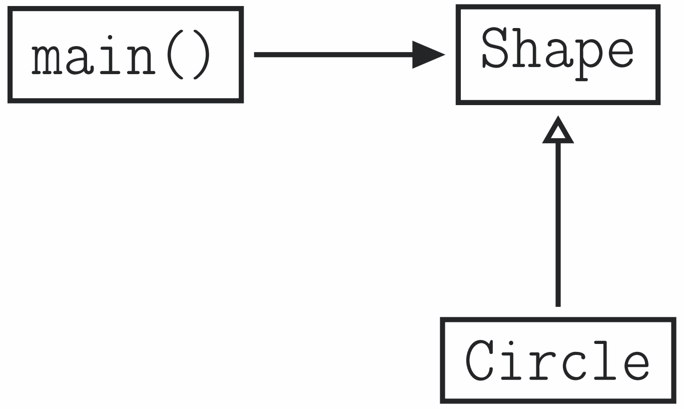
The slidedeck was presented by the author in this YouTube Video

Rational Unified Process (RUP)



Decoupling via Interfaces (in Java)

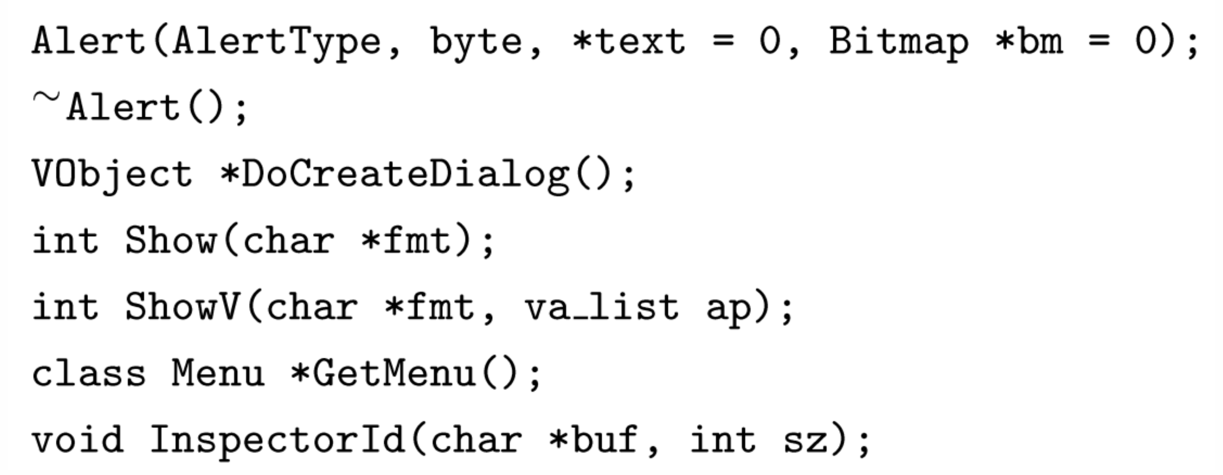
```
interface Shape
  double area();

class Circle implements Shape
  int r;
  @Override
  double area()
    return r * r * 3.14d;

void main(Shape s)
  double a = s.area();
```



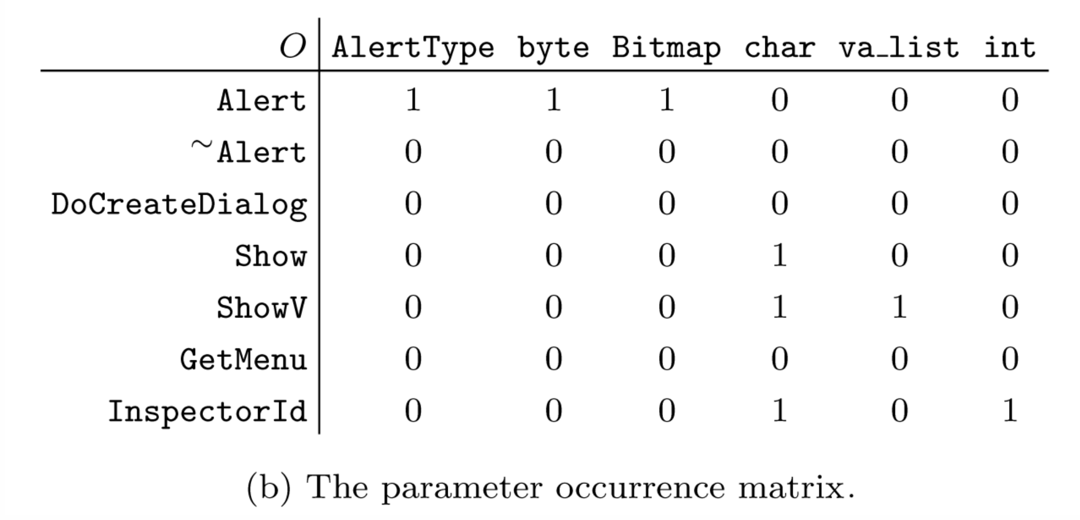
CAMC

"Cohesion Among Methods of Classes (CAMC) evaluates the relatedness of methods in the interface of a class using the parameter lists defined for the methods. It can be applied earlier in the development than can traditional cohesiveness metrics because it relies only on method prototypes declared in a class."

— *A Class Cohesion Metric for Object-Oriented Designs*, Jagdish Bansiya, Letha Etzkorn, Carl Davis and Wei Li, Journal of Object-Oriented Programming, vol. 11, no. 8, 1999



```
Alert(AlertType, byte, *text = 0, Bitmap *bm = 0);
~Alert();
VObject *DoCreateDialog();
int Show(char *fmt);
int ShowV(char *fmt, va_list ap);
class Menu *GetMenu();
void InspectorId(char *buf, int sz);
```



O	AlertType	byte	Bitmap	char	va_list	int
Alert	1	1	1	0	0	0
~Alert	0	0	0	0	0	0
DoCreateDialog	0	0	0	0	0	0
Show	0	0	0	1	0	0
ShowV	0	0	0	1	1	0
GetMenu	0	0	0	0	0	0
InspectorId	0	0	0	1	0	1

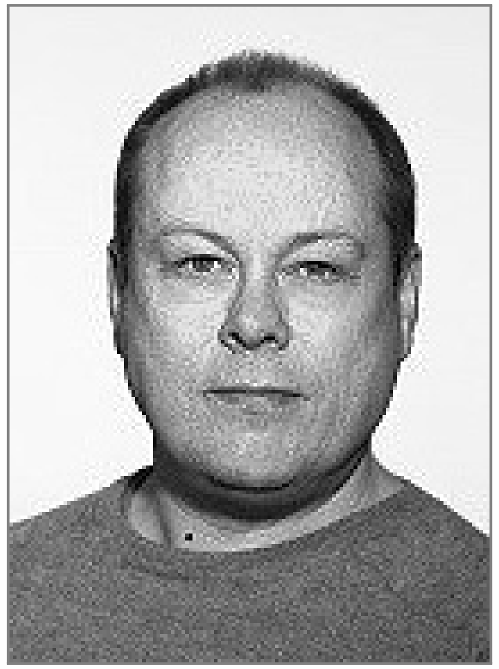
(b) The parameter occurrence matrix.

k — total number of methods

l — total number of types

$$\mathtt{CAMC} = \frac{1}{k \times l} \times \sum_{i=1}^{k} \sum_{j=1}^{l} o_{ij}$$

NHD



"The hamming distance (HD) provides a measure of disagreement between rows in a binary matrix. The Normalised Hamming Distance (NHD) metric measures agreement between rows in a binary matrix."

— *The Interpretation and utility of three cohesion metrics for object-oriented design*, Steve Counsell, Stephen Swift, Jason Crampton, ACM Transactions on Software Engineering and Methodology (TOSEM), vol. 15, no. 2, 2006

```
Alert(AlertType, byte, *text = 0, Bitmap *bm = 0);
~Alert();
VObject *DoCreateDialog();
int Show(char *fmt);
int ShowV(char *fmt, va_list ap);
class Menu *GetMenu();
void InspectorId(char *buf, int sz);
```

O	AlertType	byte	Bitmap	char	va_list	int
Alert	1	1	1	0	0	0
~Alert	0	0	0	0	0	0
DoCreateDialog	0	0	0	0	0	0
Show	0	0	0	1	0	0
ShowV	0	0	0	1	1	0
GetMenu	0	0	0	0	0	0
InspectorId	0	0	0	1	0	1

(b) The parameter occurrence matrix.

A	Alert	~Alert	DoCreateDialog	Show	ShowV	GetMenu
~Alert	3					
DoCreateDialog	3	6				
Show	2	5	5			
ShowV	1	4	4	5		
GetMenu	3	6	6	5	4	
InspectorId	1	4	4	5	4	4
	13	25	19	15	8	4

(c) The parameter agreement matrix.

k — total number of methods

l — total number of types

$$\text{NHD} = \frac{2}{l \times k \times (k-1)} \times \sum_{j=1}^{k-1} \sum_{i=j+1}^{k} a_{ij}$$

Interface Segregation Principle

"Classes that have 'fat' interfaces are classes whose interfaces are not cohesive. In other words, the interfaces of the class can be broken up into groups of methods."

— *Agile Software Development, Principles, Patterns, and Practices*, Robert C. Martin, 2002

InputStream in Java

Bad:

```
abstract class InputStream
  int read();
  int read(byte[] b);
  int read(byte[] b, int o, int l);

class FileInputStream
  implements InputStream
  native int read();
  native int read(byte[] b, int o, int l);
  int read(byte[] b)
    return read(b, 0, b.length);
```

Better (but slower!):

```
interface InputStream {
  int read(byte[] b, int o, int l);

class FileInputStream
  implements InputStream
  native int read(byte[] b, int o, int l);

class OneByteStream
  InputStream s;
  int read()
    byte[] b = new byte[1];
    s.read(b, 0, 1);
    return (int) b[0];
```

Source: Why InputStream Design Is Wrong (2016)

Also Known As...

- “interface” in Java
- “protocol” in Objective-C
- “interface” in C#
- “abstract class” in C++
- absent in Python
- absent in JavaScript
- “interface” in Go
- “trait” in Rust

Read this:

A Class Cohesion Metric for Object-Oriented Designs, Jagdish Bansiya, Letha Etzkorn, Carl Davis and Wei Li, Journal of Object-Oriented Programming, vol. 11, no. 8, 1999

The Interpretation and utility of three cohesion metrics for object-oriented design, Steve Counsell, Stephen Swift, Jason Crampton ACM Transactions on Software Engineering and Methodology (TOSEM), vol. 15, no. 2, 2006

Agile Software Development, Principles, Patterns, and Practices, Robert C. Martin, 2002

Why InputStream Design Is Wrong (2016)

Fat vs. Skinny Design (2020)

Fear of Decoupling (2018)

SRP is a Hoax (2017)

References