# Coupling

#### YEGOR BUGAYENKO

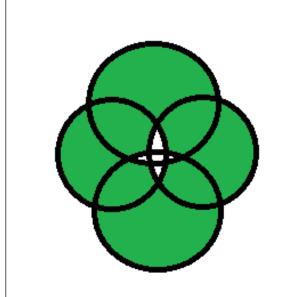
## Lecture #6 out of 24 80 minutes

All visual and text materials presented in this slidedeck are either originally made by the author or taken from public Internet sources, such as website.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



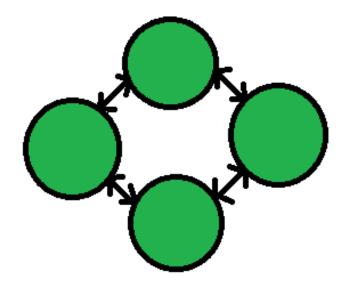
"The fewer and simpler the connections between modules, the easier it is to understand each module without reference to other modules."

Wayne P. Stevens, Glenford J. Myers, and <u>Larry L.</u>
 Constantine, *Structured Design*, IBM Systems Journal 13.2 (1974)



#### Tight coupling:

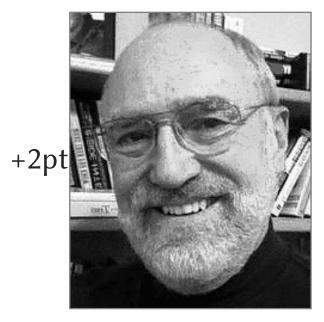
- 1. More Interdependency
- 2. More coordination
- 3. More information flow



#### Loose coupling:

- 1. Less Interdependency
- 2. Less coordination
- 3. Less information flow

Source: https://www.geeksforgeeks.org/coupling-in-java/

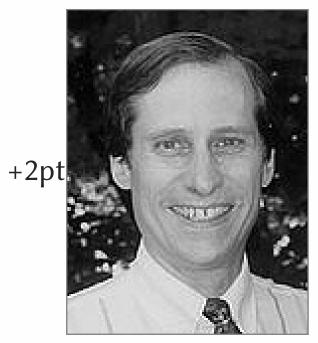


"Coupling is the measure of the strength of association established by a connection from one module to another. Strong coupling complicates a system since a module is harder to understand, change, or correct by itself if it is highly interrelated with other modules. Complexity can be reduced by designing systems with the weakest possible coupling between modules."

Wayne P. Stevens, <u>Glenford J. Myers</u>, and Larry L.
 Constantine, *Structured Design*, IBM Systems Journal
 13.2 (1974)



Source: https://www.javatpoint.com/software-engineering-coupling-and-cohesion

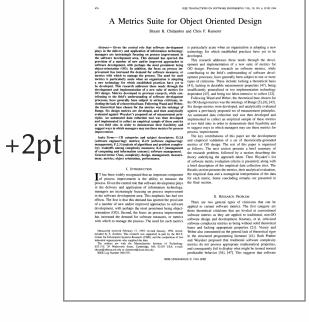


"The degree of coupling established by a particular connection is a function of several factors, and thus it is difficult to establish a simple index of coupling. Coupling depends (1) on how complicated the connection is, (2) on whether the connection refers to the module itself or something inside it, and (3) on what is being sent or received."

Wayne P. Stevens, Glenford J. Myers, and Larry L. Constantine, *Structured Design*, IBM Systems Journal 13.2 (1974)



Source: https://nordicapis.com/the-difference-between-tight-coupling-and-loose-coupling/



"Coupling Between Objects (CBO) — for a class is a count of the number of other classes to which it is coupled."

Shyam R. Chidamber and Chris F. Kemerer, A
 Metrics Suite for Object Oriented Design, IEEE
 Transactions on Software Engineering, 20.6, 1994

+2pt

1 In Trust control of the cont

IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 28, NO. 1, JANUARY 2

#### A Hierarchical Model for Object-Oriented Design Quality Assessment

lagdish Bansiya, Member, IEEE, and Carl G. Davis, Fellow, IEEE

Abstract—This paper describes an improved interrother model for the assessment of high-level design gualty articles in objectcented designs. In this mode, structural and behavioral design properties of leasts, educes, and the residentity has evaluation of the control of the properties of the control of the properties of the properti

index Terms—Quality model, quality attributes, design metrics, product metrics, object-oriented metric

1 Introduction

In demand for quality software continues to intensify due to our society's increasing dependence on software and the often devastating effect that a software error can a sew in terms of life financial loss, or time delays. Today's loss was in terms of life financial loss, or time delays. Today's loss are seen to the second of the second

measure and assure quality are far from settled issues. The switch to the object-oriented paraligm has changed to object-oriented paraligm has changed to object the paraligm of the paraligm of the paraligm Taditional software product metrics that evaluate produc characteristics used as size, complesity, performance, and quality must be changed to rely on some fundamentally different notions such as encapsulation, inheritance, and polymorphism which are inherent in object-orientation The has led to the definition of namey we metric [8], [12] [20] to measure the products of the object-oriented However, the new object-oriented metrics are varied in However, the new object-oriented metrics are varied in However, the new object-oriented metrics are varied in \$\frac{1}{2}\$.

However, the new object-oriented metrics are varied in what they measure, how they are used in measuring, and when they are applicable. Many of the newer metrics have only been validated with small, and sometimes nonrealistic

- In biologies is until the Department of Saturements and Computer Science Galifornia State University, Hayanad, CA 94542.
   E-mail: jhunsiyaribesuhayanad.adu.
   C. Darsis is with the Computer Science Department, University of Alabam in Hantsville, Hundsville, AI 35599. E-mail: dismisilies asili.adu.
- Monuscript received 24 Nov. 1997; revised 29 Nov. 1998; accepted 27 j 2000. Recommended for acceptance by D.R. Jeffery. For information on obtaining regards of this article, phase and e-mail

For information on obtaining reprints of this article, please send e-mail to: tselfcomputer.org, and reference IEEECS Log Number 105978.

effectiveness of the metrics on large complex projects such
such as those encountered in an industrial environment is no
known. Finally, if the goal is assessing the external quality
individual metrics, then there must be a well defined way o
connecting the two.

eavailable for object-oriented software analyses can be applied only after a product is complete, or nearly complete. They rely upon information extracted from the origination of the complete of the complete of the total to the lost product in the complete of the total to the lost product in the complete of the complete of the product. Thus, there is a new year for metrics and models that can be applied in the enty stage of development enquirements and design) to ensure that the analysis and design have invested internal product that the subjustic and the complete of the complete of product. This would give developers an opportunity to find problems, remove irregulatties and nonconformance to 51, standards, and eliminate unwanted complexity early in the development cycle. This should againfantly thelp in the complete of the complete of the complete of development cycle. This should againfantly thelp in the complete of the complete of productions and the implementation, as well the complete of the complete of productions and the complete of productions and the complete of productions are considered to the complete of the c

Fortmately, the object-oriented approach naturally lends itself to an endy assessment and evaluation. Object-oriented methodologies require significant effort early in the development of the end of

0009-5599(02517.00 © 2002 IEEE
d licensed use limited to: ECOLE POLYTECHNIQUE DE MONTREAL Downloaded on October 7, 2009 at 11:09 from IEEE Xpices. Restrictions apply.

"Direct Class Coupling (DCC) — this metric is a count of the different number of classes that a class is directly related to. The metric includes classes that are directly related by attribute declarations and message passing (parameters) in methods."

 Jagdish Bansiya and Carl G. Davis, A Hierarchical Model for Object-Oriented Design Quality Assessment, IEEE Transactions on Software Engineering, 28.1, 2002



"The biggest problems come from uncontrolled coupling at the <u>upper levels</u>. I don't worry about the number of modules coupled together, but I look at the pattern of dependency relationship between the modules."

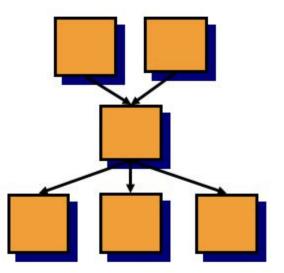
Martin Fowler, *Reducing Coupling*, IEEE Software,



"Low-to-medium fan-out means having a given class use a low-to-medium number of other classes. High fan-out (more than about seven) indicates that a class uses a large number of other classes and may therefore be overly complex. High fan-in refers to having a high number of classes that use a given class. High fan-in implies that a system has been designed to make good use of utility classes at the lower levels in the system."

- Steven McConnell, Code Complete, 2004

Fan-in = number of ingoing dependencies Fan-out = number of outgoing dependencies



Heuristic: a high fan-in/fan-out indicates a high complexity

(c) Natalia Kokash, Leiden Institute of Advanced Computer Science

#### An Evolutionary Study of Fan-in and Fan-out Metrics in OSS

A. Mubarak, S. Counsell and R.M. Hierons

Department of Information Systems and Computing, Brunel University

Lisheiden, UK, Email: steve counsell@brunel.ac.uk

Administration of the properties of the properti

Keywords-coupling, Java, fan-in, fan-out, package.

+2pt

I. INTRODUCTION

the Object-Oriented (OO) community that executive coupling between classes creates a level of complicity that can complicate subsequent maintenance and represents a 'street' of "maintenance problem." I practice, a class that is highly of "maintenance problem." I practice, a class that is highly complicating or removal from the system to mitigate both complicating or removal from the system to mitigate both current and potential future problems. A problem that immediately attent however for the developer when immediately attent however for the developer when the complication of the compliance of the complication of the so, then are those coupling depredendencies." If the so, then are those coupling depredendencies "I motion," is more difficult to coupling, as the three forces of the compliance In this paper, we investigate versions of five Open Sourcesystem (CSS) Societing on two well-known coupling metric "fair-in" (i.e., incoming coupling) and "fan-out" (i.e., outgoing outping). We used an uniternated tool to extract each of the outping metrics from those five systems. The research used to be used to the coupling and the coupling outping outping coupling naturally have low outgoing coupling and second, does this relationship women over time? In other ords, does the potential maintenance problem become worst terms of fair-in and fan-out values?

II. MOTIVATION AND RELATED WORK

The reusers his this paper is motionated by a number of fusion. Trendy, previous reasonal [15] has shown that here is a trade freely, previous reasonal [15] has also mit have the in a trade coupling through imported packages and the introduction of the introduction of the previous reasonal properties of the properties of production of the properties of the properties of the properties of production of the properties of the properties of the properties of production of the properties of the properties of production of the properties of the properties of production of the properties of properties of the properties of production of the properties of previous research [16] which showed that the fasion and fast out metrics tended to be reliably by mall of cleans remove and/or fased may be difficult to move or remove from yours. This question has imported further examination or youter. This question has imported further examination or provious research that the properties of youter. This question has imported further examination or youter. This question has imported further examination or youter. This question has imported further examination or youter. The properties has been also as well as well as a proper provious research to the provious properties of the provious research that the provious properties are provious research to the provious properties of the provious research provious provious provious properties and the provious research provious provious properties are the provious research provious provio

In terms of related work, the research presented relates to are of software evolution, coupling metrics and the use of of software evolution, coupling metrics and the use of Lehrana [: provide the backdrop for many past evolutionary studie Evolution has also been the subject of simulation studies [! in and this has allowed OSS evolution to be studied in contrasting way to that empirically. The research presented

"We also found evidence of certain 'key' classes (with both high fan-in and fan-out) and 'client' and 'server'-type classes with just high fan-out and fan-in, respectively."

 A. Mubarak et al., An Evolutionary Study of Fan-in and Fan-out Metrics in OSS, Proceedings of the 4th International Conference on Research Challenges in Information Science (RCIS), 2010

Fan-out, as a metric, is supported by a few tools:

• Checkstyle for Java

 $\bullet$  <u>CCCC</u> for C++, C, and Java

• module-coupling-metrics for Python



"Afferent coupling (denoted by  $\mathbf{Ca}$ ) is a metric that indicates the total number of other projects/boundaries that are dependent upon it. Efferent coupling (denoted by  $\mathbf{Ce}$ ) is another metric that is the verse of Afferent Coupling. It is the total number of projects that a given project depends on. Instability another metric that is a ratio:  $\mathbf{I} = \mathbf{Ce}/(\mathbf{Ce} + \mathbf{Ca})$ . This metric is a ratio between 0 and 1. With 0 meaning it's totally stable and 1 meaning it's unstable."

 Derek Comartin, Write Stable Code using Coupling Metrics, 2021

## Types of Coupling (some of them)

- Content Coupling is when one module modifies or relies on the internal workings of another module (e.g., accessing local data of another module).
- Global Coupling is when two modules share the same global data (e.g., a global variable).
- External Coupling occurs when two modules share an externally imposed data format, communication protocol, or device interface.
- <u>Control Coupling</u> is one module controlling the flow of another, by passing it information on what to do (e.g., passing a what-to-do flag).
- Stamp Coupling is when modules share a composite data structure and use only a part of it, possibly a different part (e.g., passing a whole record to a function that only needs one field of it).

- <u>Data Coupling</u> is when modules share data through, for example, parameters. Each datum is an elementary piece, and these are the only data shared (e.g., passing an integer to a function that computes a square root).
- Message Coupling can be achieved by state decentralization (as in objects) and component communication is done via parameters or message passing (see Message passing).
- <u>Subclass Coupling</u> describes the relationship between a child and its parent. The child is connected to its parent, but the parent isn't connected to the child.
- Temporal Coupling is when two actions are bundled together into one module just because they happen to occur at the same time.

#### Source:

https://wiki.edunitas.com/IT/en/114-10/Coupling-(computer-programming)\_1430\_eduNitas.html

## Fear of Decoupling

```
interface Money {
  double cents();
}

void send(Money m) {
  double c = m.cents();
  // Send them over via the API...
}

class OneDollar implements Money {
  @Override
  double cents() {
  return 100.0d;
  }
}
```

```
class EmployeeHourlyRate
implements Money {
    @Override
    double cents() {
        // Fetch the exchange rate;
        // Update the database;
        // Calculate the hourly rate;
        // Return the value.
    }
}
```

"Polymorphism makes sofware more fragile ... to make it more robust!"

## Temporal Coupling

## Tight coupling (not good):

```
List<String> list =
new LinkedList<>();
Foo.append(list, "Jeff");
Foo.append(list, "Walter");
return list;
```

### Loose coupling (good):

```
return Foo.with(
Foo.with(
new LinkedList<>(),
"Jeff"
),
"Walter"
);
```

https://www.yegor256.com/2015/12/08/temporal-coupling-between-method-calls.html

## Distance of Coupling

```
class Temperature {
  private int t;
  public String toString() {
    return String.format("%d F", this.t);
  }
}

Temperature x = new Temperature();
String txt = x.toString();
String[] parts = txt.split(" ");
int t = Integer.parseInt(parts[0]);
```

"The larger the number (or the mean of all numbers), the worse the design: in good design we are not supposed to take something out of a method and then do some complex processing. The distance metric will tell us exactly that: how many times, and by how much, we violated the principle of loose coupling."

https://www.yegor256.com/2020/10/27/distance-of-coupling.html

## Read this:

Structured Design, Wayne P. Stevens, et al., IBM Systems Journal, 13.2, 1974

A Hierarchical Model for Object-Oriented Design Quality Assessment, Jagdish Bansiya et al., IEEE Transactions on Software Engineering, 28.1, 2022

An Overview of Various Object Oriented Metrics, Brij Mohan Goel et al., International Journal of Information Technology & Systems, 2.1, 2014

Analysing the Contribution of Coupling Metrics for the Development and Management of Process Architectures, Daniel Braunnagel et al., ECIS, 2015

New Metric: the Distance of Coupling (2020)

Fear of Decoupling (2018)

Reflection Means Hidden Coupling (2022)