2nd hardest thing in computer science

@pawel_lewtak

Definition?

There are only two hard things in Computer Science: cache invalidation and naming things.

Phil Karlton

There are 2 hard problems in computer science: cache invalidation, naming things, and off-by-1 errors

Leon Bambrick

There are only two hard problems in distributed systems:

2. Exactly-once delivery

1. Guaranteed order of messages

2. Exactly-once delivery

Mathias Verraes

#2 Naming things*

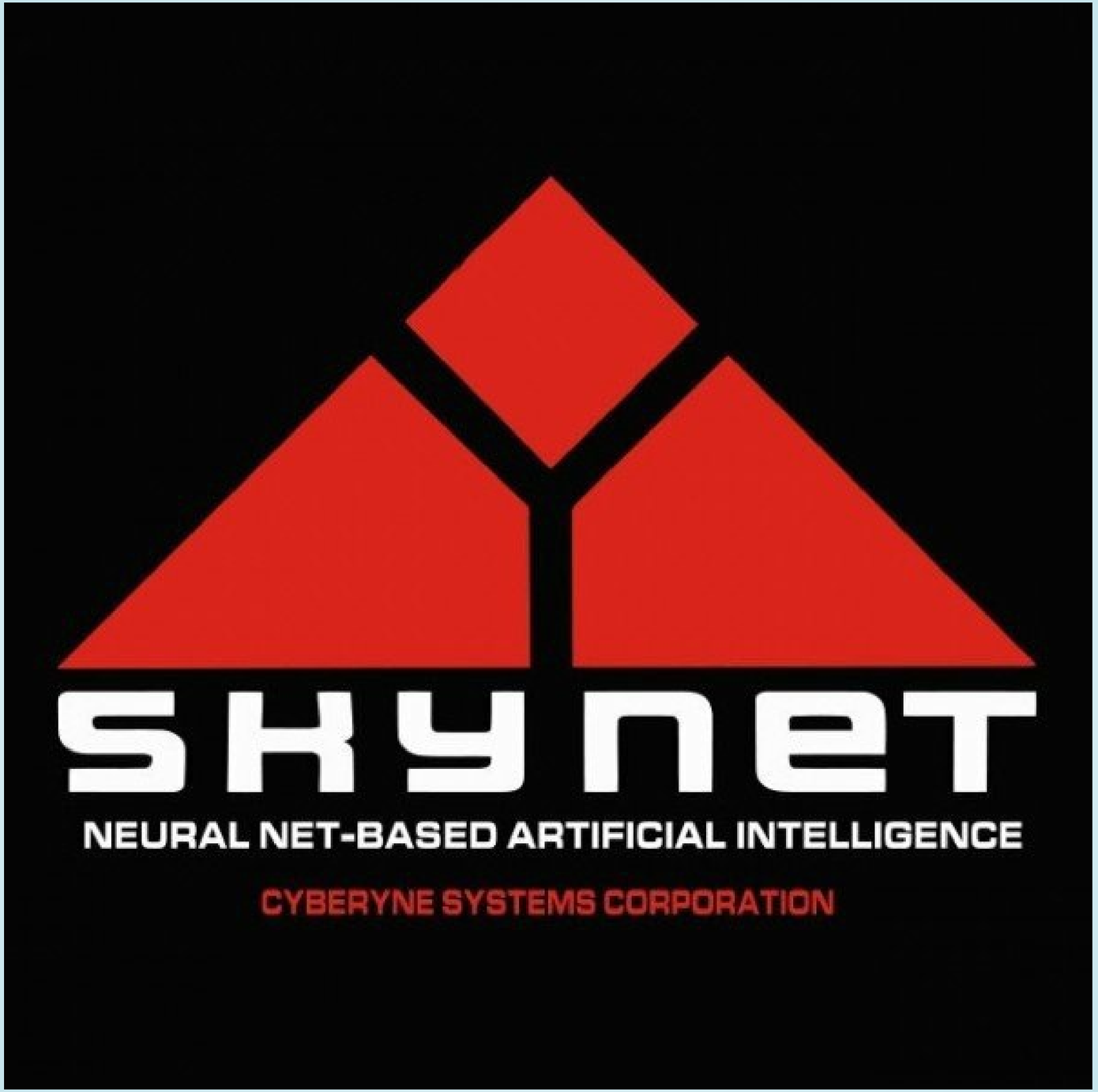
*things

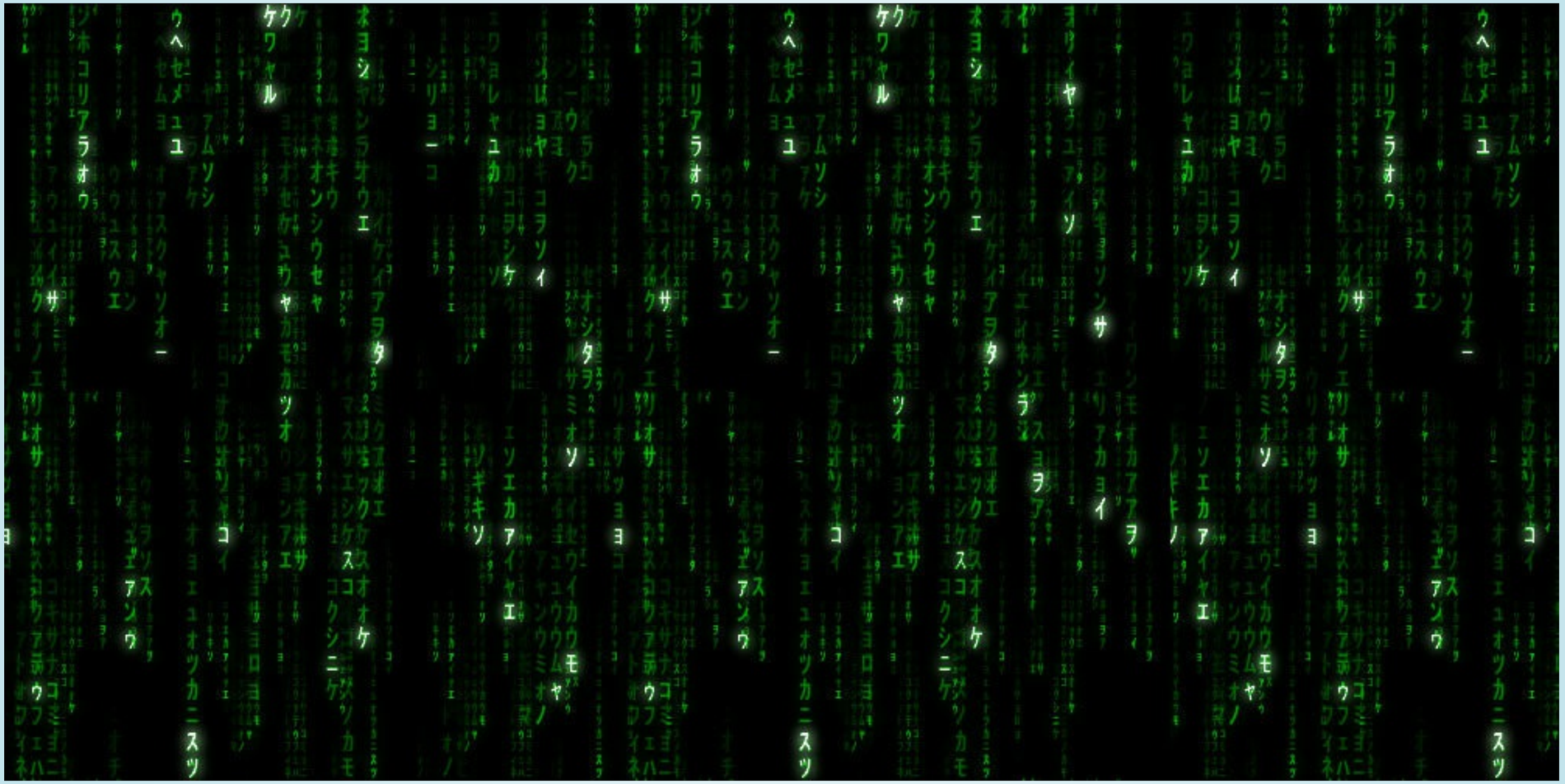
- variables
- methods
- classes
- modules
- comments
- inline docs

You don't code for CPU

You don't code for interpreter

You don't code for compiler


SKYNET
NEURAL NET-BASED ARTIFICIAL INTELLIGENCE
CYBERYNE SYSTEMS CORPORATION

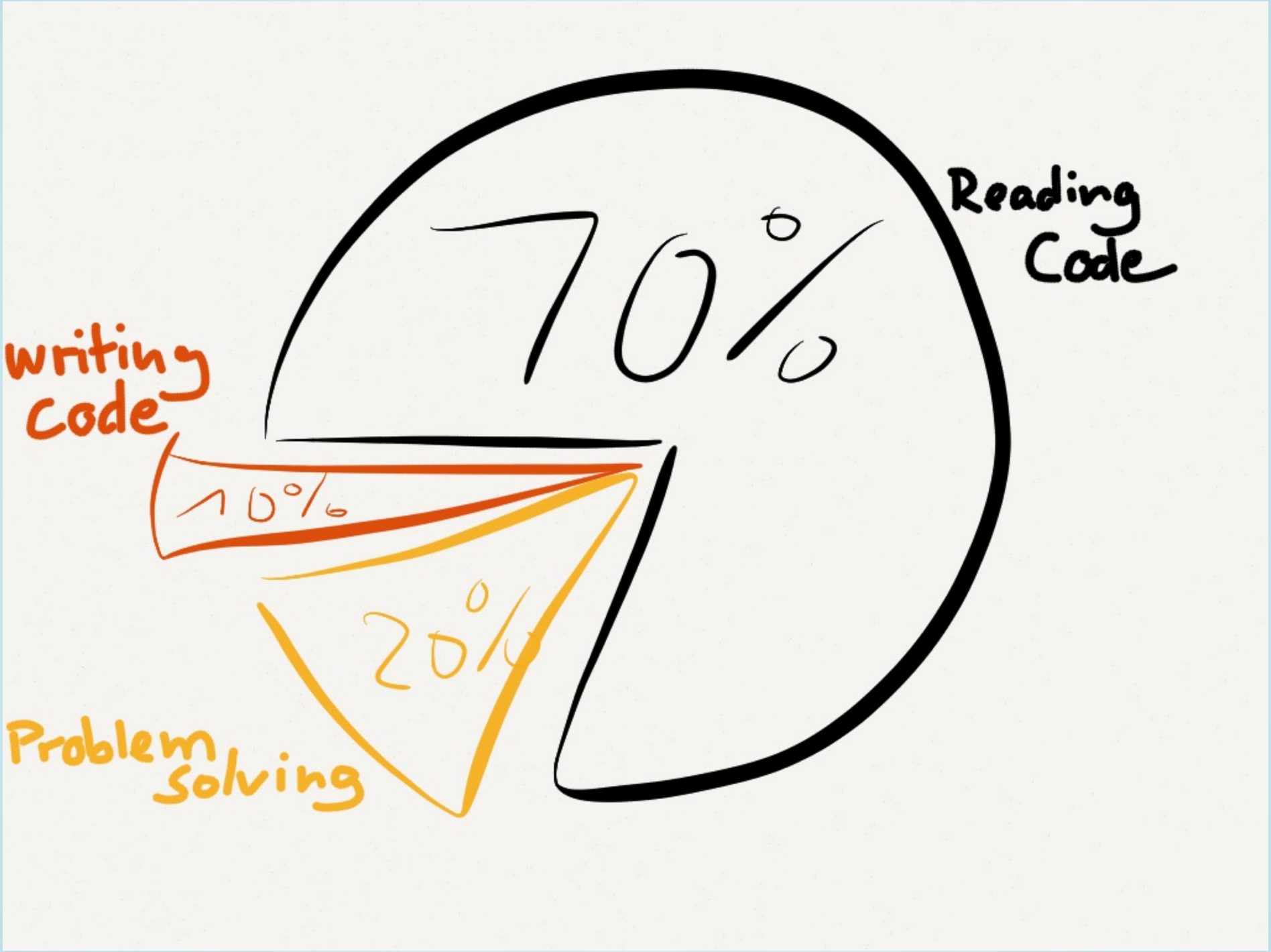


You code for people

You code for other developers

You code for your future self

Don't code,
wrote prose


70%
Reading Code
writing code
10%
20%
Problem solving

Always code as if the guy who ends up maintaining your code will be a violent psychopath who knows where you live.

Code for readability.

John F. Woods

Comprehension

~70%

```
def a(b):
    c = sorted(b)
    d = len(b)
    if d % 2 == 1:
        return c[(d - 1) / 2]
    else:
        return (c[d/2 - 1] + c[d/2]) / 2
```

```
def median(pool):
    copy = sorted(pool)
    size = len(copy)
    if size % 2 == 1:
        return copy[(size - 1) / 2]
    else:
        return (copy[size/2 - 1] + copy[size/2]) / 2
```

Self-documenting code

Code written by somebody else

Programming is mapping

- from problem domain
- via intermediate domain
- into programming domain

DDD FTW

Worst variable name

data


Thank You,
Captain
Obvious!

Second worst name?

data2

```
total = price * qty
total2 = total - discount
total2 += total * taxrate

total3 = purchase_order_value + available_credit
if total2 < total3:
  print ("You can't afford this order.")
```

```
order_total = price * qty
payable_total = order_total - discount
payable_total += payable_total * taxrate

available_funds = purchase_order_value + availble_credit
if payable_total < available_funds:
  print ("You can't afford this order.")
```

No-one sets out to write legacy code

Rachel Willmer

Broken window theory

Code will decay

Design patterns

Misapplied Java design patterns

are the root of all
AbstractWordFactoryFactory("evil")

HN comment

Naming conventions

TL;DR

- CamelCaseClass
- methodName
- someVariable
- CAPITAL_CONSTANT

syntax < semanthics

Common issues

Pseudo getter

getData()

with extra operations inside

getCreateObject()

- fetch
- find
- lookup
- create
- calculate

Not really a boolean

isActive()

```
def isActive():
  if cond:
    return 'false'
  return 'true'
```

isValid()

```
def isValid():
  if inputIsValid:
    return True
```

Plural / singular names

```
def getPerson():
  return ['John Doe', 'Jane Doe']

def getEmployers():
  return 'John Doe'
```

Misleading docs

```
def getLowestPrice(user):
  pass
```

```
def getLowestPrice(user):
  """Actually it returns the highest price."""
  pass
```

More than one responsibility

Abbreviations

pos
mod
abs
auth

Synonyms

<ThatThing>Manager

- UserManager
- StringManager
- ProductManager
- etc.

Alternatives

- Builder
- Writer
- Adapter
- Factory
- Handler
- Provider
- Converter

Magic numbers

```
import requests

response = requests.get('https://2017.confitura.pl/')
if response.status_code == 200:
  print ("It works!")
elif response.status_code == 418:
  print ("Unexpected teapot!")
```

```
import requests

response = requests.get('https://2017.confitura.pl/')
if response.status_code == requests.codes.ok:
  print ("It works!")
elif response.status_code == requests.codes.teapot:
  print ("Unexpected teapot!")
```

Useless comments

```
def getData():
  """ Returns the data. """
  pass

def getMaxIdFromDb():
  """ Return maximum ID value from the database."""
  pass
```

Explain why,
not what or how

Bad name:

- Does more that what is says
- Says more than what it does
- Does the opposite
- Contains more than what it says
- Says more than what it contains
- Contains the opposite

Good practices

Specific names

No generics

Short names

Do not use negation

isNotEnabled()

Consistent names

Single responsibility

Domain terms

Think about it

ASCII only

~~Hungarian notation~~

```
hostList, hostSet => hosts, validHosts
valueString => firstName, lowercasedSKU
intNumber => accountNumber
```

Tests!

How?

Agree on standards

Practice

Improve vocabulary

Refactor

Code reviews

Short, bite size, single logical change

Code ownership

Team leader's role

Links

- https://groups.google.com/d/msg/comp.lang.c++/rYCO5yn4lXw/oITtSkZOtoUJ
- https://www.cqse.eu/publications/2005-concise-and-consistent-naming.pdf
- http://www.cs.loyola.edu/~lawrie/papers/lawrieJese07.pdf
- https://www.researchgate.net/publication/224079441_Relating_Identifier_Naming_Flaws_and_Code_Quality_An_Empirical_Study
- http://www.veneraarnaoudova.com/wp-content/uploads/2014/10/2014-EMSE-Arnaodova-et-al-Perception-LAs.pdf
- http://archive.oreilly.com/pub/post/the_worlds_two_worst_variable.html
- https://twitter.com/amokleben/status/868377283496751104?s=09
- https://twitter.com/tmmx/status/865308678903267328
- https://trustartist.com/2015/01/27/pair-programming-economics/

Thank you!

@pawel_lewtak