Code Coverage and Cyclomatic Complexity

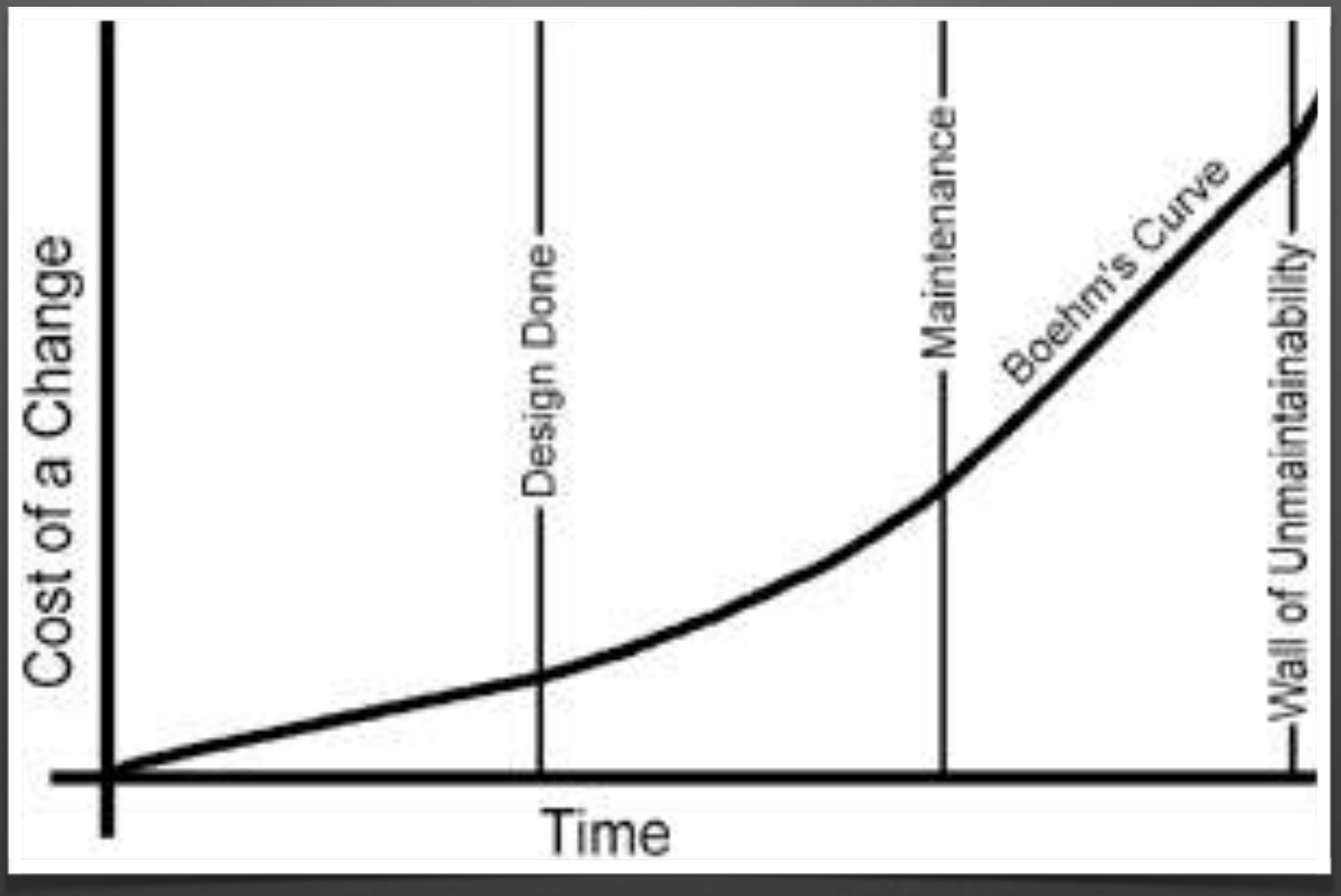
Arnan Maipradit
Kanin Sirisith
Nutnicha Charoenporn
Thai Pangsakulyanont

Source Code QA

“A software project is only as good as its QA workflow..”

–Esprima Unit Test Page
http://esprima.org/test/index.html

Source Code QA



Idea: make it as easy as possible to change code

http://www.agile-process.org/change.html

Software Rot

Refactoring

Technical Debts

http://martinfowler.com/bliki/TechnicalDebt.html

Hard Way

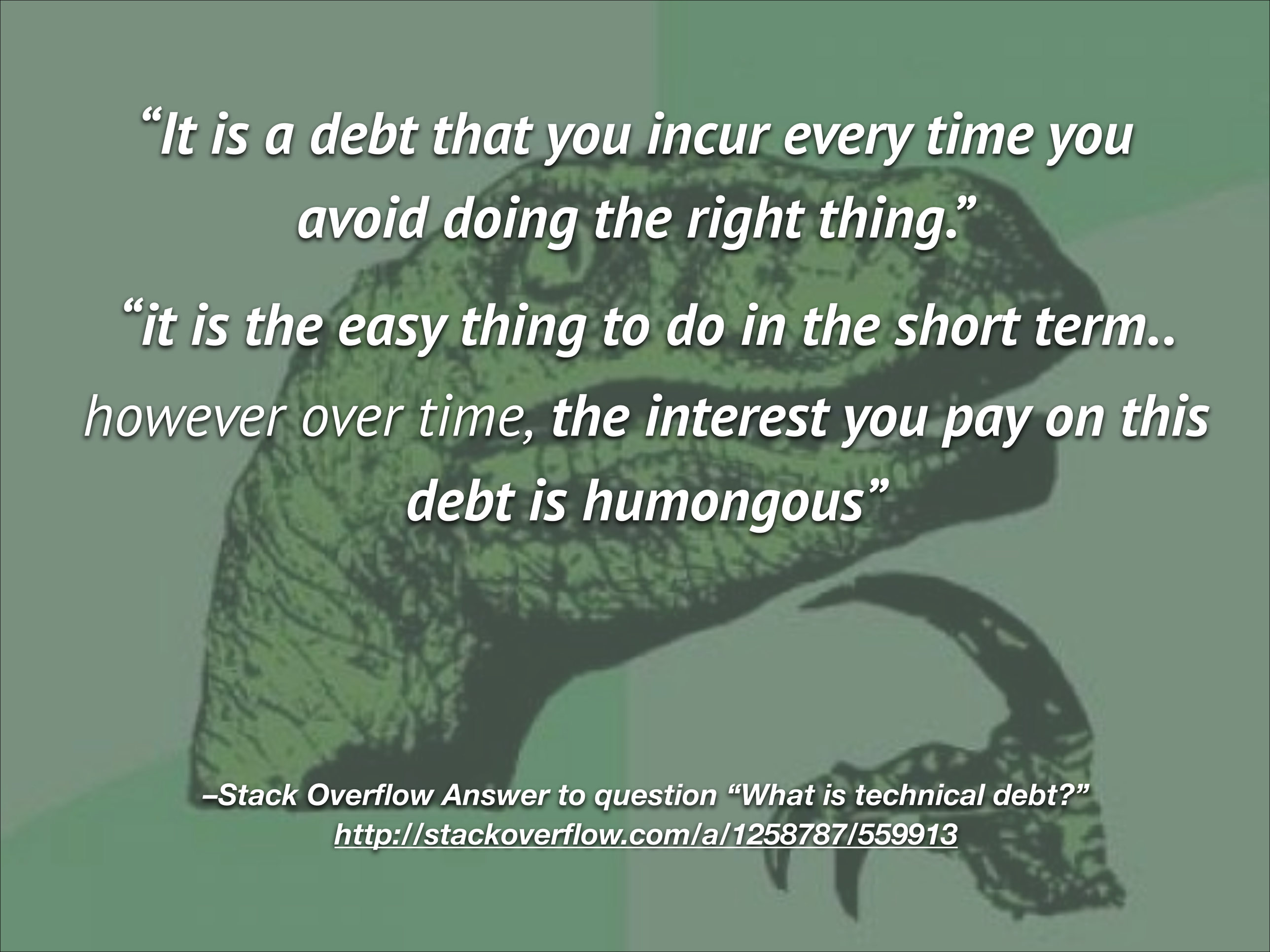
The Right Way

Easy Way

Quick and Dirty

“It is a debt that you incur every time you avoid doing the right thing.”

–Stack Overflow Answer to question “What is technical debt?”
http://stackoverflow.com/a/1258787/559913

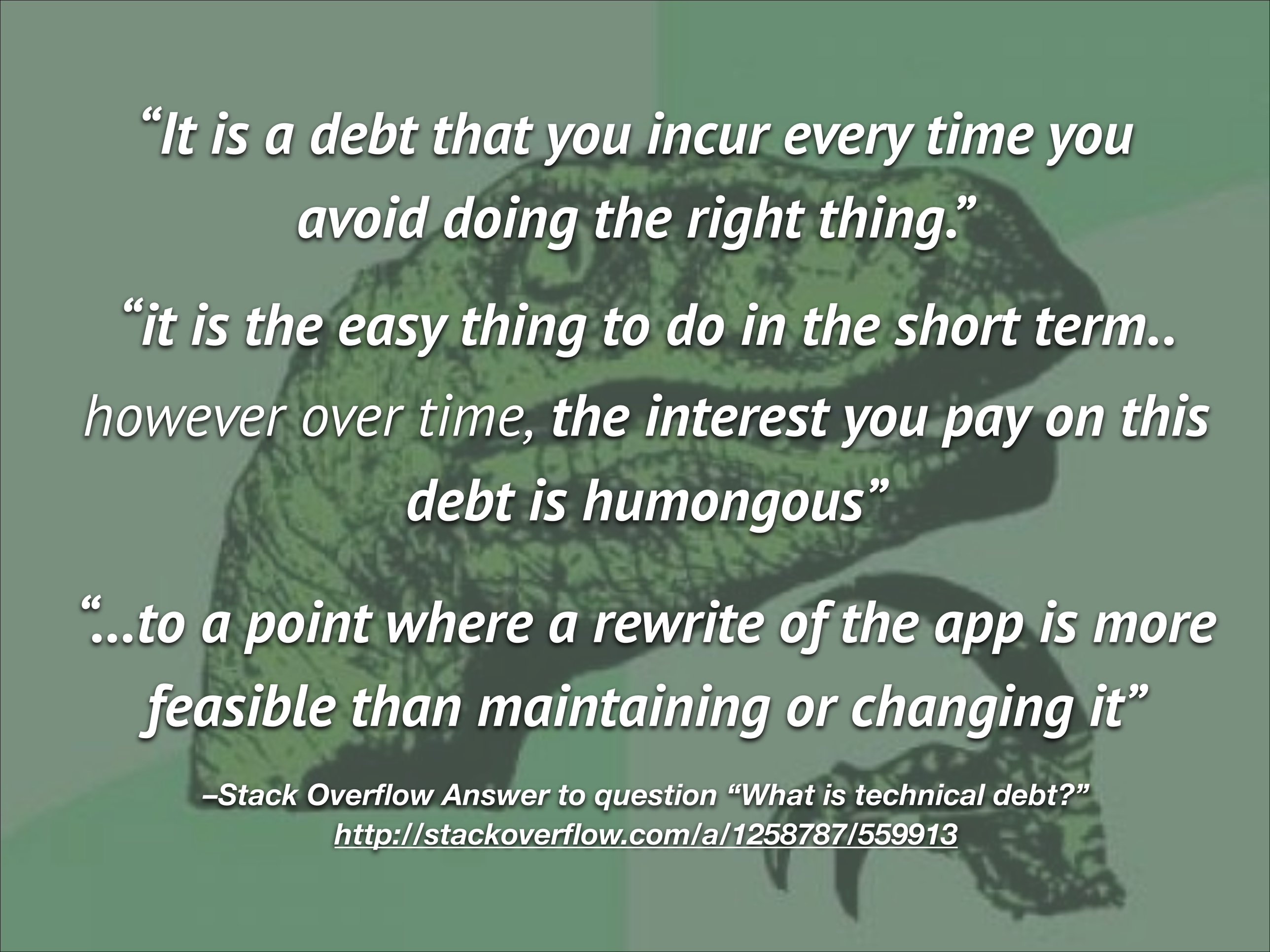


“It is a debt that you incur every time you avoid doing the right thing.”

“it is the easy thing to do in the short term..

however over time, ***the interest you pay on this debt is humongous”***

–Stack Overflow Answer to question “What is technical debt?”
http://stackoverflow.com/a/1258787/559913



“It is a debt that you incur every time you avoid doing the right thing.”

“it is the easy thing to do in the short term..

however over time, ***the interest you pay on this debt is humongous”***

“...to a point where a rewrite of the app is more feasible than maintaining or changing it”

–Stack Overflow Answer to question “What is technical debt?”
http://stackoverflow.com/a/1258787/559913

```
function getKey(key) {
  if (key == 'S') return 0;
  if (key == 'D') return 1;
  if (key == 'F') return 2;
  if (key == 'SPACE') return 3;
  if (key == 'J') return 4;
  if (key == 'K') return 5;
  if (key == 'L') return 6;
  return null;
}
```

Implement Key Mapping System

```
function getKey(key) {
  if (key == 'S' || key == 'Q') return 0;
  if (key == 'D' || key == 'W') return 1;
  if (key == 'F' || key == 'E') return 2;
  if (key == 'SPACE') return 3;
  if (key == 'J' || key == 'U') return 4;
  if (key == 'K' || key == 'I') return 5;
  if (key == 'L' || key == 'O') return 6;
  return null;
}
```

More complexity → Harder to change

Fix one bug, another bug appears!

More subtle! Harder to find!
More expensive to fix!

Fix one bug, another bug appears!

“[Y]ou can not effectively increase software quality at the end of the project by testing and fixing bugs. It remains low quality but with fewer bugs.”

–Don Wells, Surprise! Software rots!, 2009.
http://www.agile-process.org/change.html

How to keep software quality high?

Measure them!

Code Quality Measures

(a.k.a. software metrics)

Code Quality Measures

(a.k.a. software metrics)

code coverage

churn

flog

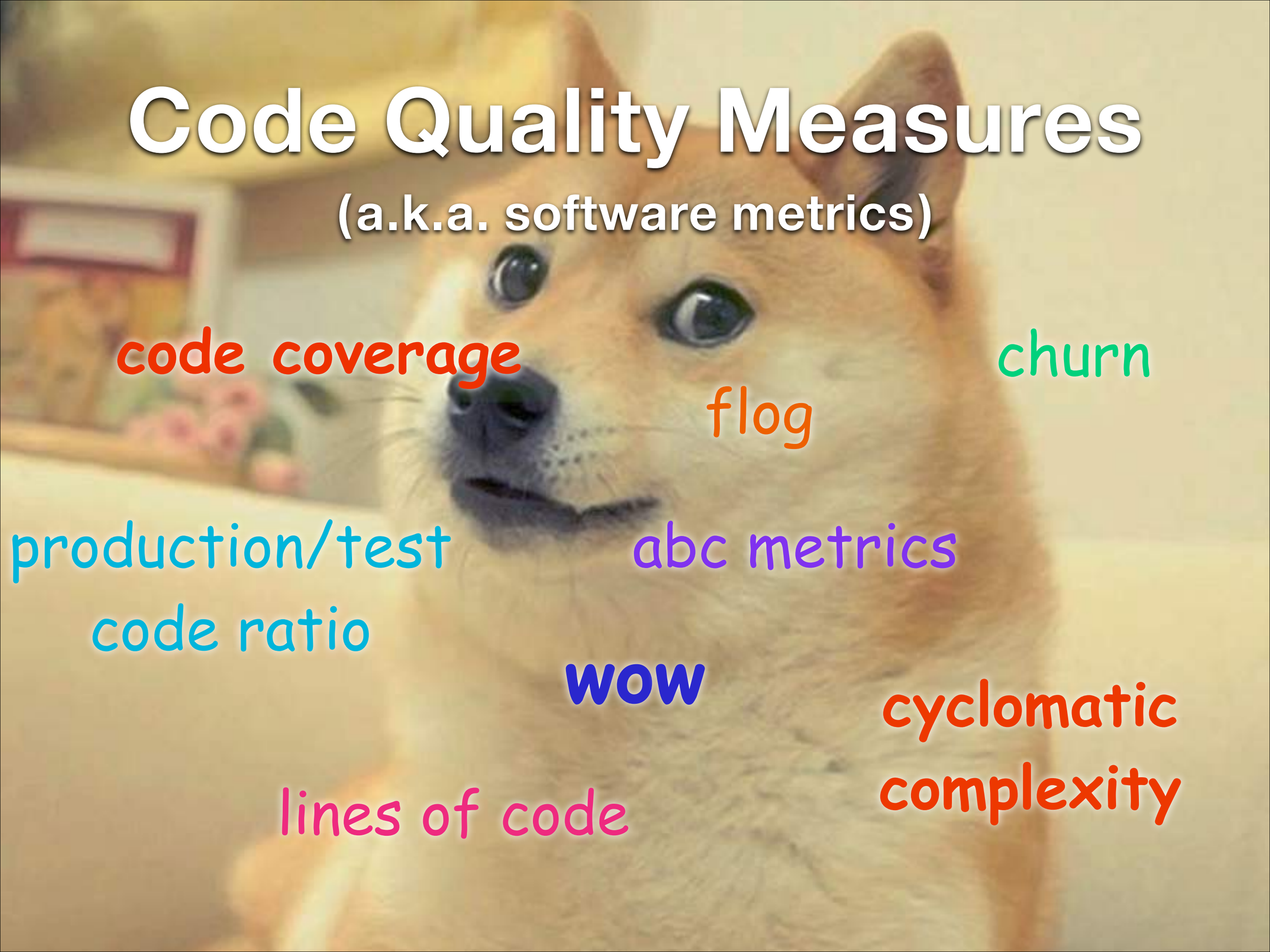
production/test code ratio

abc metrics

wow

cyclomatic complexity

lines of code



Code Coverage

% of code “covered” by test

Ensure that your tests are actually testing your code

More code coverage

More code coverage

More thoroughly tested

More code coverage

More thoroughly tested

Lower chance of bugs

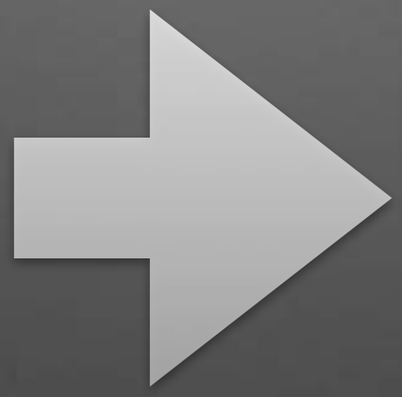
How does it work?

Instrumentation

```
function max(a, b) {
  if (a > b) {
    return a;
  } else {
    return b;
  }
}
```

Instrumentation

```
function max(a, b) {
  if (a > b) {
    return a;
  } else {
    return b;
  }
}
```



```
function max(a,b) {
  check.f['1']++; check.
  if (a > b) {
    check.b['1'][0]++;
    check.s['3']++;
    return a;
  } else {
    check.b['1'][1]++;
    check.s['4']++;
    return b;
  }
}
```

Run (“exercise”) the instrumented code

(e.x. unit test, functional test, …)

Run (“exercise”) the instrumented code

(e.x. unit test, functional test, …)

Report is generated by the tool

Code coverage report for **esprima/esprima.js**

Statements: **99.72%** (1760 / 1765)

Branches: **98.49%** (1107 / 1124)

Functions: **100%** (154 / 154)

Lines: **99.72%** (1760 / 1765)

Code coverage report for **esprima/esprima.js**

Statements: **99.72%** (1760 / 1765) Branches: **98.49%** (1107 / 1124)

Functions: **100%** (154 / 154) Lines: **99.7**

```
39
40    1 (function (root, factory) {
41    1     'use
42
43          // Universal Module Definition (UMD) to support AMD, CommonJ
44          // Rhino, and plain browser loading.
45    1     I if (typeof define === 'function' && define.amd) {
46              define(['exports'], factory);
47    1     } else E if (typeof exports !== 'undefined') {
```

branch not covered

if path not taken

statement not covered

else path not taken

Code coverage report for **esprima/esprima.js**

Statements: **99.72%** (1760 / 1765)

Branches: **98.49%** (1107 / 1124)

Functions: **100%** (154 / 154)

Lines: **99.72%** (1760 / 1765)

Code coverage report for **esprima/esprima.js**

Statements: **99.72%** (1760 / 1765)

Branches: **98.49%** (1107 / 1124)

Functions: **100%** (154 / 154)

Lines: **99.72%** (1760 / 1765)

WHAT?!

Coverage Types

Function Coverage

```
function max(a, b) {
  if (a > b) {
    return a;
  } else {
    return b;
  }
}

function min(a, b) {
  if (a < b) {
    return a;
  } else {
    return b;
  }
}
```

Function Coverage

```
function max(a, b) {
  if (a > b) {
    return a;
  } else {
    return b;
  }
}
```

```
function min(a, b) {
  if (a < b) {
    return a;
  } else {
    return b;
  }
}
```

Statement Coverage (C0)

```
function max(a, b) {
  if (a > b) {
    return a;
  } else {
    return b;
  }
}

function min(a, b) {
  if (a < b) {
    return a;
  } else {
    return b;
  }
}
```

Statement Coverage (C0)

```
function max(a, b) {
  if (a > b) {
    return a;
  } else {
    return b;
  }
}

function min(a, b) {
  if (a < b) {
    return a;
  } else {
    return b;
  }
}
```

Branch Coverage (C1)

```
function binary_search(a, l, r, v) {
  if (!r) {
    r = a.length - 1;
  } // else
  var m = Math.floor((l + r) / 2);
  if (a[m] == v) {
    return m;
  } else if (a[m] > v) {
    return binary_search(a, l, r - 1, v);
  } else {
    return binary_search(a, l + 1, r, v);
  }
}
```

Branch Coverage (C1)

```
function binary_search(a, l, r, v) { ✔
  if (!r) { ✔
    r = a.length - 1;
  } // else ✔
  var m = Math.floor((l + r) / 2);
  if (a[m] == v) { ✔
    return m;
  } else if (a[m] > v) { ✔
    return binary_search(a, l, r - 1, v);
  } else { ✔
    return binary_search(a, l + 1, r, v);
  }
}
```

Branch Coverage (C1)

```
if (f(mid) && (mid == 0 || !f(mid - 1))) {
```

Branch Coverage (C1)

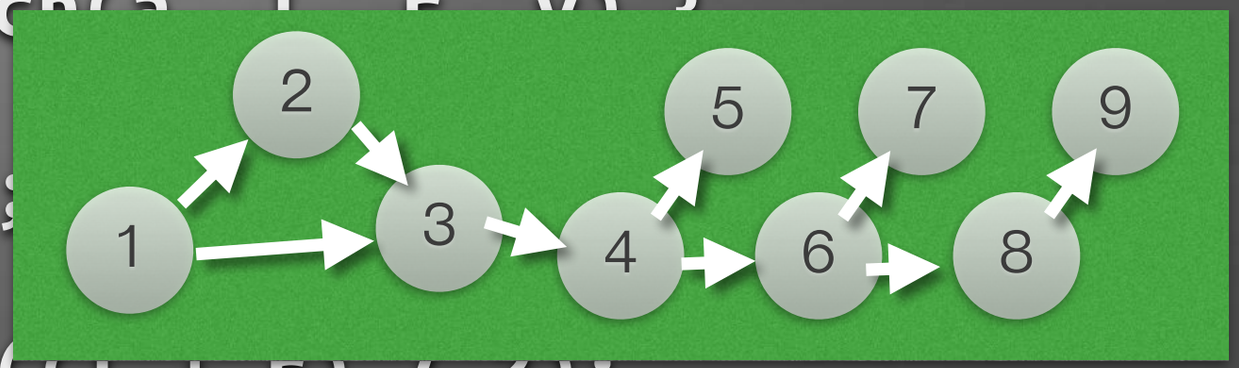
```
if (f(mid) && (mid == 0 || !f(mid - 1))) {
```

Path Coverage (C2)

```
   function binary_search(a, l, r, v) {
(1)  if (!r) {
(2)    r = a.length - 1;
     }
(3)  var m = Math.floor((l + r) / 2);
(4)  if (a[m] == v) {
(5)    return m;
(6)  } else if (a[m] > v) {
(7)    return binary_search(a, l, r - 1, v);
(8)  } else {
(9)    return binary_search(a, l + 1, r, v);
     }
   }
```

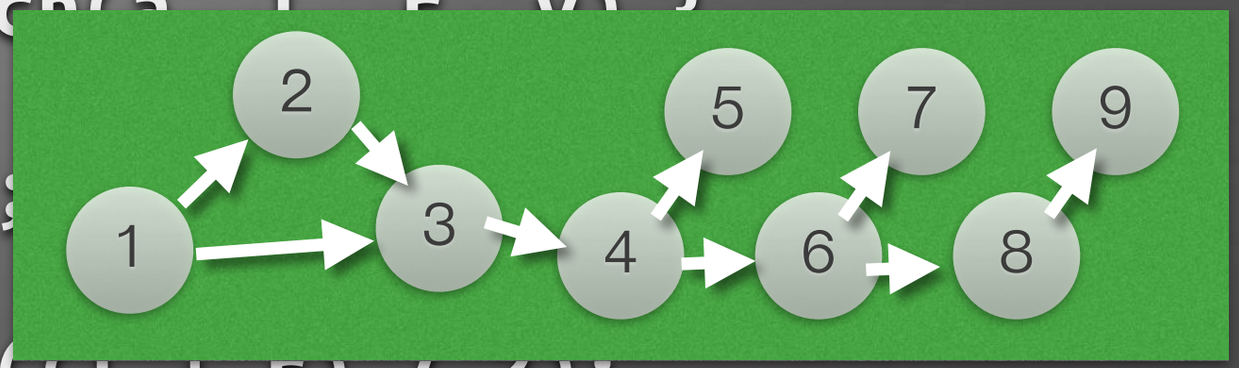
Path Coverage (C2)

```
   function binary_search(a, l, r, v) {
(1)  if (!r) {
(2)    r = a.length - 1;
     }
(3)  var m = Math.floor((l + r) / 2);
(4)  if (a[m] == v) {
(5)    return m;
(6)  } else if (a[m] > v) {
(7)    return binary_search(a, l, r - 1, v);
(8)  } else {
(9)    return binary_search(a, l + 1, r, v);
     }
   }
```



Path Coverage (C2)

```
   function binary_search(a, l, r, v) {
(1)  if (!r) {
(2)    r = a.length - 1;
     }
(3)  var m = Math.floor((l + r) / 2);
(4)  if (a[m] == v) {
(5)    return m;
(6)  } else if (a[m] > v) {
(7)    return binary_search(a
(8)  } else {
(9)    return binary_search(a
     }
   }
```



Possible paths

- ✔ 1→2→3→4→5
- ✔ 1→2→3→4→6→7
- ✔ 1→2→3→4→6→8→9
- ✔ 1→3→4→5
- ✔ 1→3→4→6→7
- ✔ 1→3→4→6→8→9

Path Coverage (C2)

```
   function binary_search(a, l, r, v) {
(1)  if (!r) {
(2)    r = a.length - 1;
     }
(3)  var m = Math.floor((l + r) / 2);
(4)  if (a[m] == v) {
(5)    return m;
(6)  } else if (a[m] > v) {
(7)    return binary_search(a, l, r - 1, v);
(8)  } else {
(9)    return binary_search(a, l + 1, r, v);
     }
   }
```

Possible paths

- ✔ 1→2→3→4→5

How much code coverage?

No clear cut answer..

*"**100% test coverage** is a natural side effect of proper development practices, and therefore a **bare minimum indicator of quality.**"*

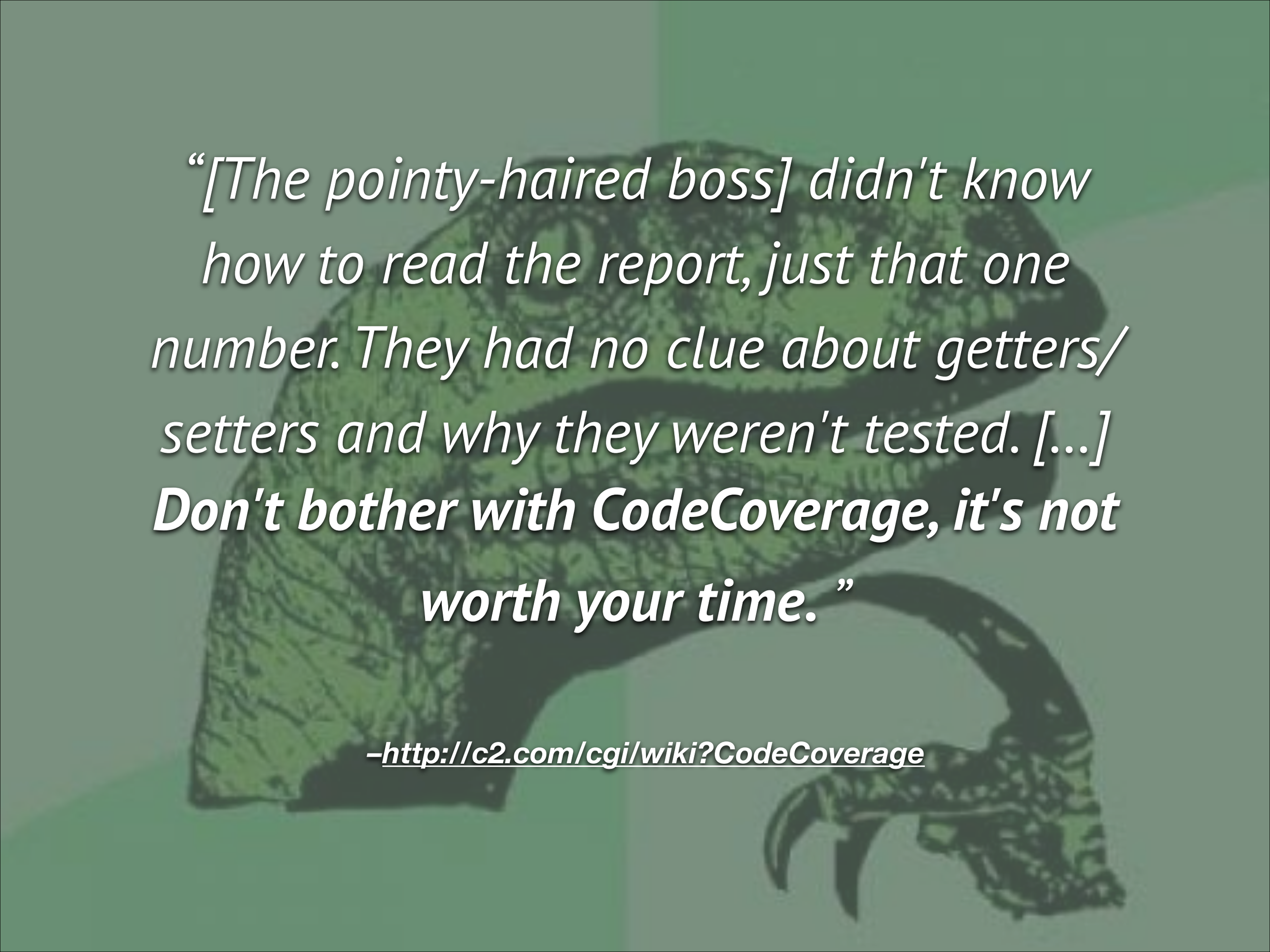
–Robert Martin (Uncle Bob) , *quoted from Deciphering Ruby Code Metrics*
http://blog.codeclimate.com/blog/2013/08/07/deciphering-ruby-code-metrics/

“Seven don’ts of testing:

1. Don’t aim for 100% coverage.”

–David Heinemeier Hansson *(creator of Ruby on Rails), Testing like the TSA.*

http://37signals.com/svn/posts/3159-testing-like-the-tsa



“[The pointy-haired boss] didn't know how to read the report, just that one number. They had no clue about getters/setters and why they weren't tested. [...] ***Don't bother with CodeCoverage, it's not worth your time.”***

–http://c2.com/cgi/wiki?CodeCoverage

Let’s ask StackOverflow

Unit testing code coverage - do you have 100% coverage?

Do your unit tests constitute 100% code coverage? Yes or no, and why or why not.

unit-testing code-coverage

edited Sep 25 '09 at 5:29

womp

67.3k 13 137 191

asked Sep 25 '09 at 5:01

Chris

7,805 14 75 125

–http://stackoverflow.com/q/1475520

No for several reasons :

- It is really expensive to reach the 100% coverage, compared to the 90% or 95% for a benefit that is not obvious.
- Even with 100% of coverage, your code is **not** perfect. Take a look at this method (in fact it depends on which type of coverage you are talking about - *branch coverage*, *line coverage*...):

edited Jul 11 '10 at 16:53

Robert Harvey ♦

95.5k ● 16 ● 146 ● 258

answered Sep 25 '09 at 5:18

romaintaz

28.2k ● 13 ● 103 ● 170

No, because there is a practical trade-off between perfect unit tests and actually finishing a project :)

edited Sep 25 '09 at 5:23

Benjol

14.8k ● 7 ● 78 ● 156

answered Sep 25 '09 at 5:03

Andrew Hare

146k ● 22 ● 343 ● 451

add comment

Yes we do.

It depends on what language and framework you're using as to how easy that is to achieve though.

We're using Ruby on Rails for my current project. Ruby is very *"mockable"* in that you can stub/mock out large chunks of your code without having to build in overly complicated class composition and construction designs that you would have to do in other languages.

That said, we only have 100% *line* coverage (basically what rcov gives you). You still have to think about testing all the required branches.

This is only really possible if you include it from the start as part of your continuous integration build, and **break the build** if coverage drops below 100% - prompting developers to immediately fix it. Of course you could choose some other number as a target, but if you're starting fresh, there isn't much difference for the effort to get from 90% to 100%

answered Sep 25 '09 at 5:53

madlep

12.4k 5 27 46

Yes we do.

It depends on what language and framework you're using as to how easy that is to achieve though.

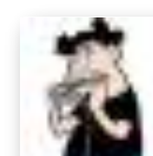
We're using Ruby on Rails for my current project. Ruby is very *"mockable"* in that you can stub/mock out large chunks of your code without having to build in overly complicated class composition and construction designs that you would have to do in other languages.

testing all the required branches.

break the build if coverage drops below 100% - prompting developers to immediately fix it. Of course you

"If you know you have to have 100% test coverage, you stop writing overly complex if/else/while/try/ catch monstrosities and Keep It Simple Stupid."

answered Sep 25 '09 at 5:53



madlep

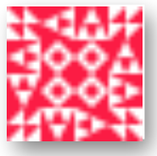
12.4k • 5 • 27 • 46

To all the 90% coverage tester:

The problem with doing so is that the 10% hard to test code is also the not-trivial code that contains 90% of the bug! This is the conclusion I got empirically after many years of TDD.

And after all this is pretty straightforward conclusion. This 10% hard to test code, is hard to test **because** it reflect tricky business problem or tricky design flaw or both. These exact reasons that often leads to buggy code.

answered Feb 20 '11 at 10:19



Patrick from NDepend tea

6,388 25 32

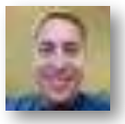
What is a reasonable code coverage % for unit tests (and why)?

If you were to mandate a minimum percentage code-coverage for unit tests, perhaps even as a requirement for committing to a repository, what would it be?

Please explain how you arrived at your answer (since if all you did was pick a number, then I could have done that all by myself ;)

unit-testing code-coverage

edited Jun 26 '09 at 6:49



Quinn Taylor

27k ●6 ●77 ●101

asked Sep 18 '08 at 4:25

sanity

10.8k ●13 ●67 ●134

add comment

–http://stackoverflow.com/q/90002

*The great master pointed at a pot of boiling water and said: "**How many grains of rice should I put in that pot?**"*

–Alberto Savoia, *Testivus On Test Coverage.*
http://stackoverflow.com/a/90021/559913

Demonstration

Cyclomatic Complexity

… measures the “complexity”

More complexity

More complexity

Harder to maintain

More complexity

Harder to maintain
test
debug
change
understand

Code Coverage
need to run code

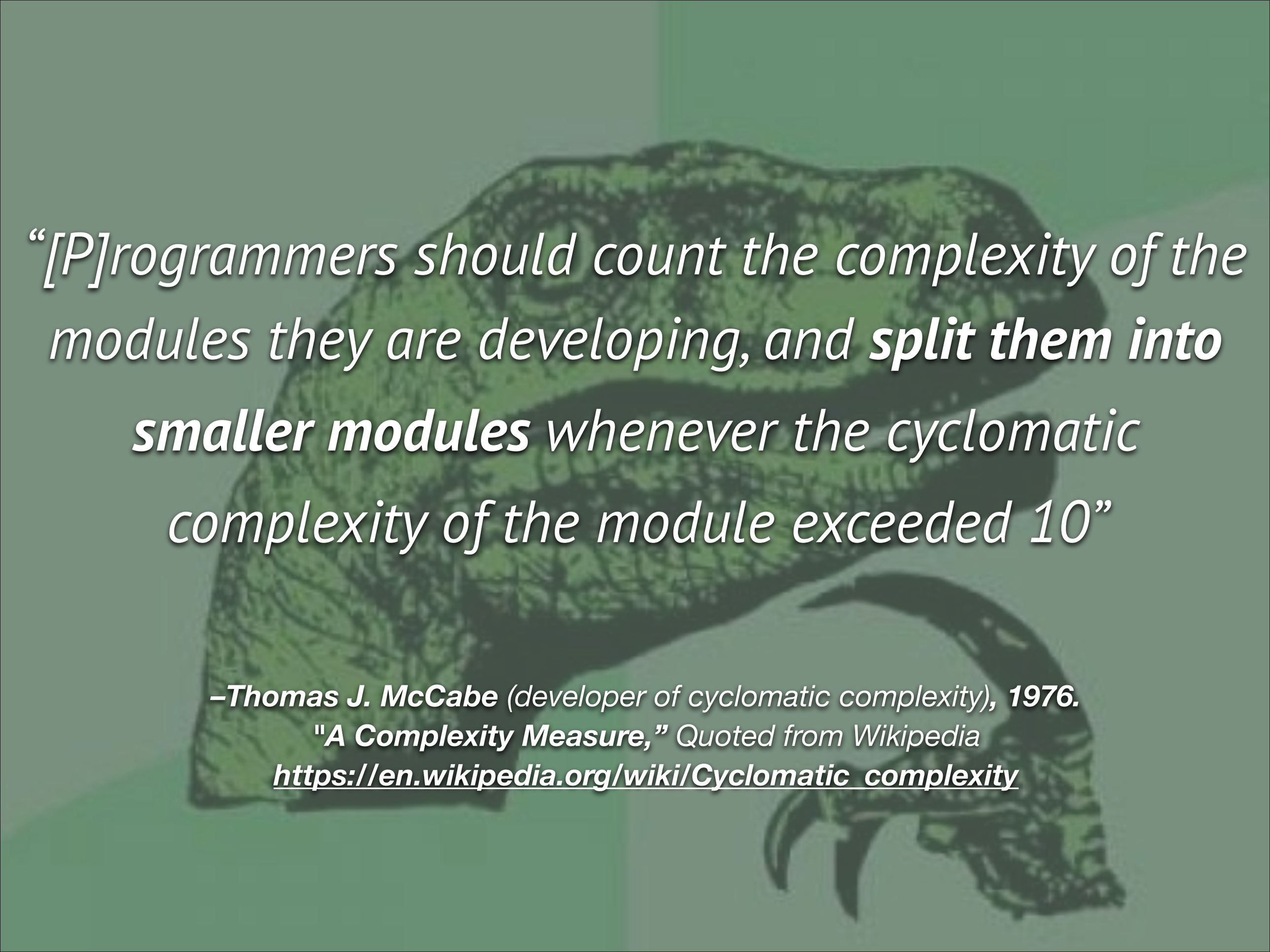
Cyclomatic Complexity
no need to run code
(called “static analysis”)

… measures number of

“linearly independent paths”

through a program’s source code

How to use the number?



*"[P]rogrammers should count the complexity of the modules they are developing, and **split them into smaller modules** whenever the cyclomatic complexity of the module exceeded 10"*

–Thomas J. McCabe *(developer of cyclomatic complexity)**, 1976.***
"A Complexity Measure," *Quoted from Wikipedia*
https://en.wikipedia.org/wiki/Cyclomatic_complexity

102?

*"For each module, either **limit** cyclomatic complexity to [the agreed-upon limit] or provide a written **explanation** of why the limit was exceeded."*

–McCabe, Watson, 1996. "Structured Testing: A Testing Methodology Using the Cyclomatic Complexity Metric," *Quoted from Wikipedia*
https://en.wikipedia.org/wiki/Cyclomatic_complexity

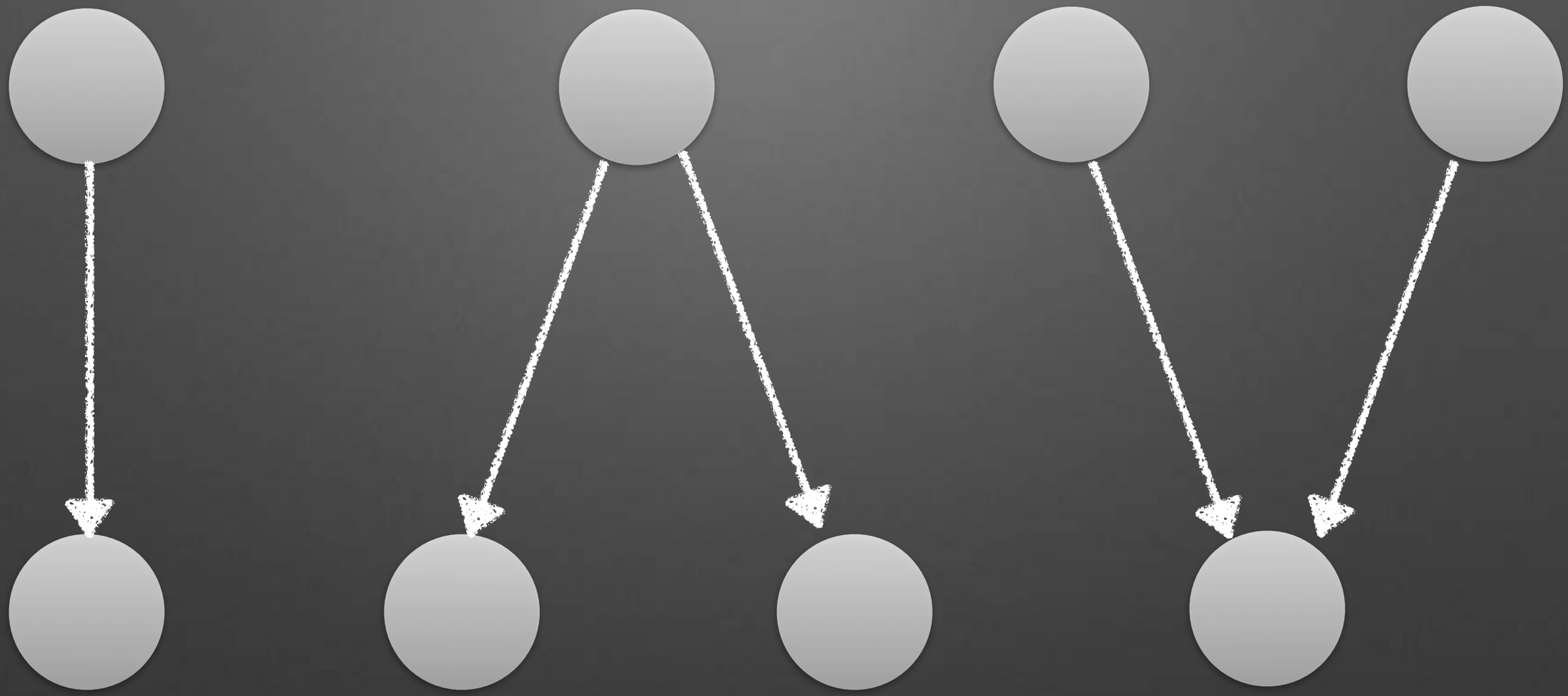
Demonstration

How to Calculate?

Code

Control Flow Graph

Sequential Flow



Branch

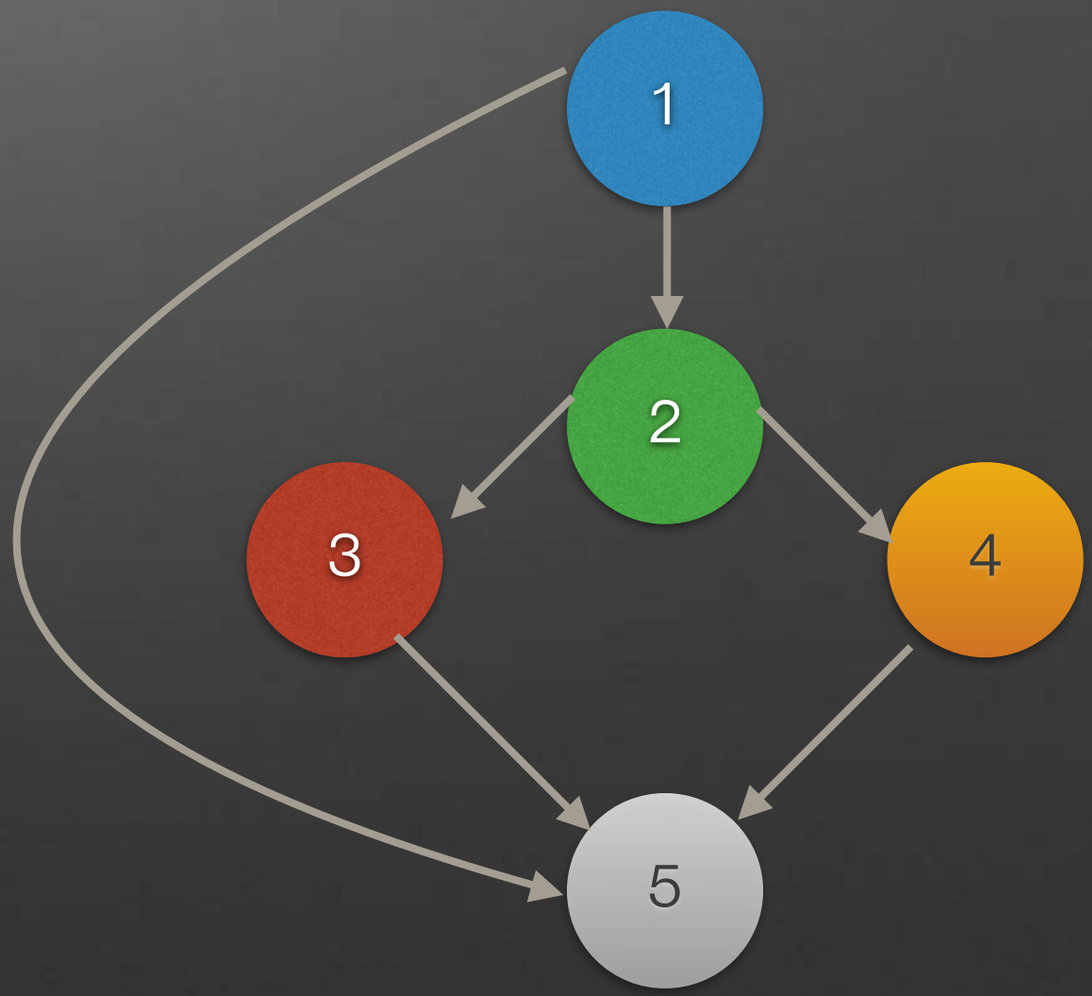
Merge

If-Else Clause

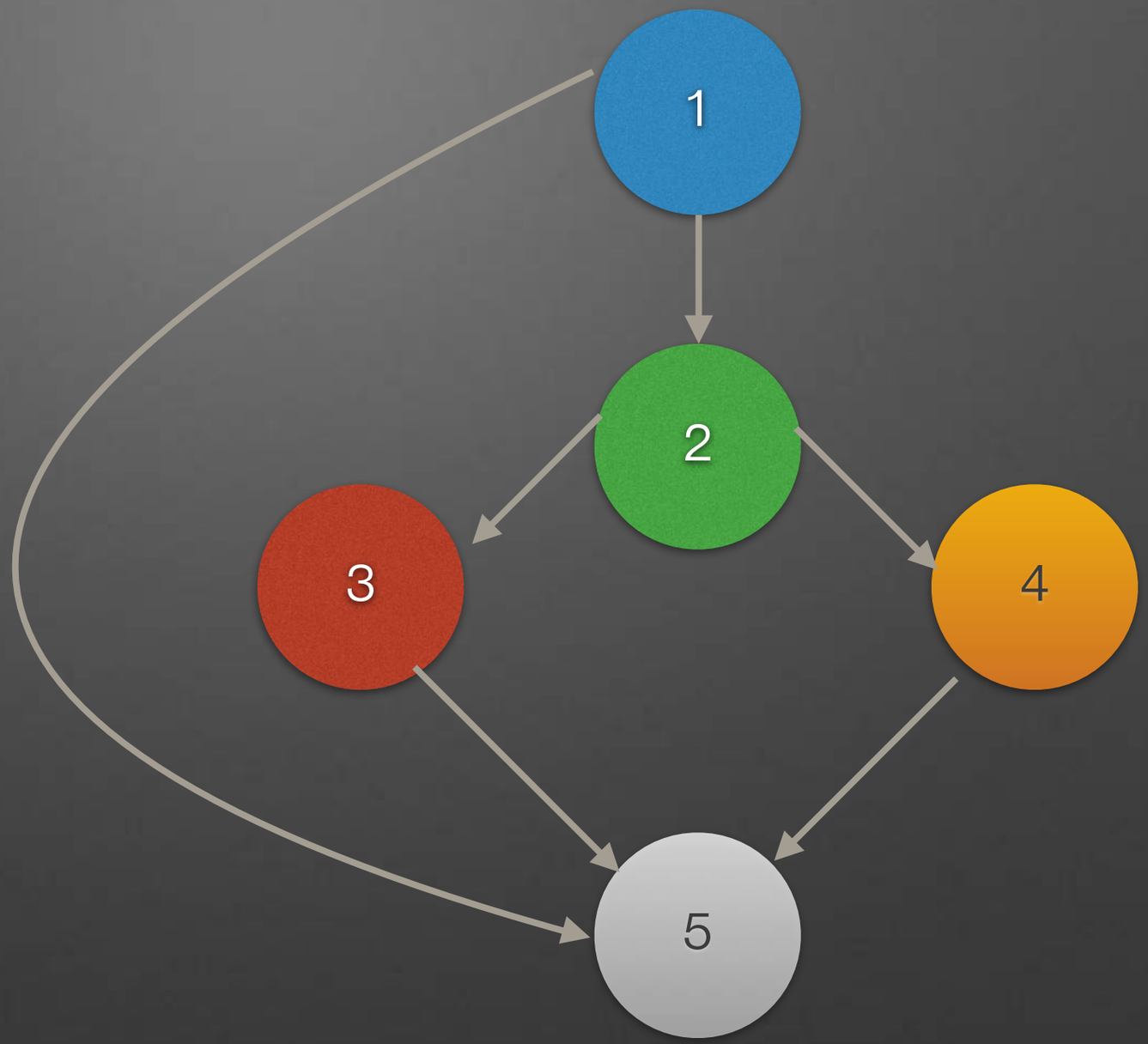
```
function insertion_procedure(int b, int c) {
(1) if (a == 111) {
(2)    if (b > c) {
(3)       a = b;
       } else {
(4)       a = c;
       }
    }
(5) Console.WriteLine(a);
}
```

If-Else Clause

```
function insertion_procedure(int b, int c) {
(1) if (a == 111) {
(2)    if (b > c) {
(3)       a = b;
       } else {
(4)       a = c;
       }
    }
(5) Console.WriteLine(a);
}
```



If-Else Clause



6 edges

5 nodes

1 connected component

Calculation

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

Calculation

Number of Edges

Nodes

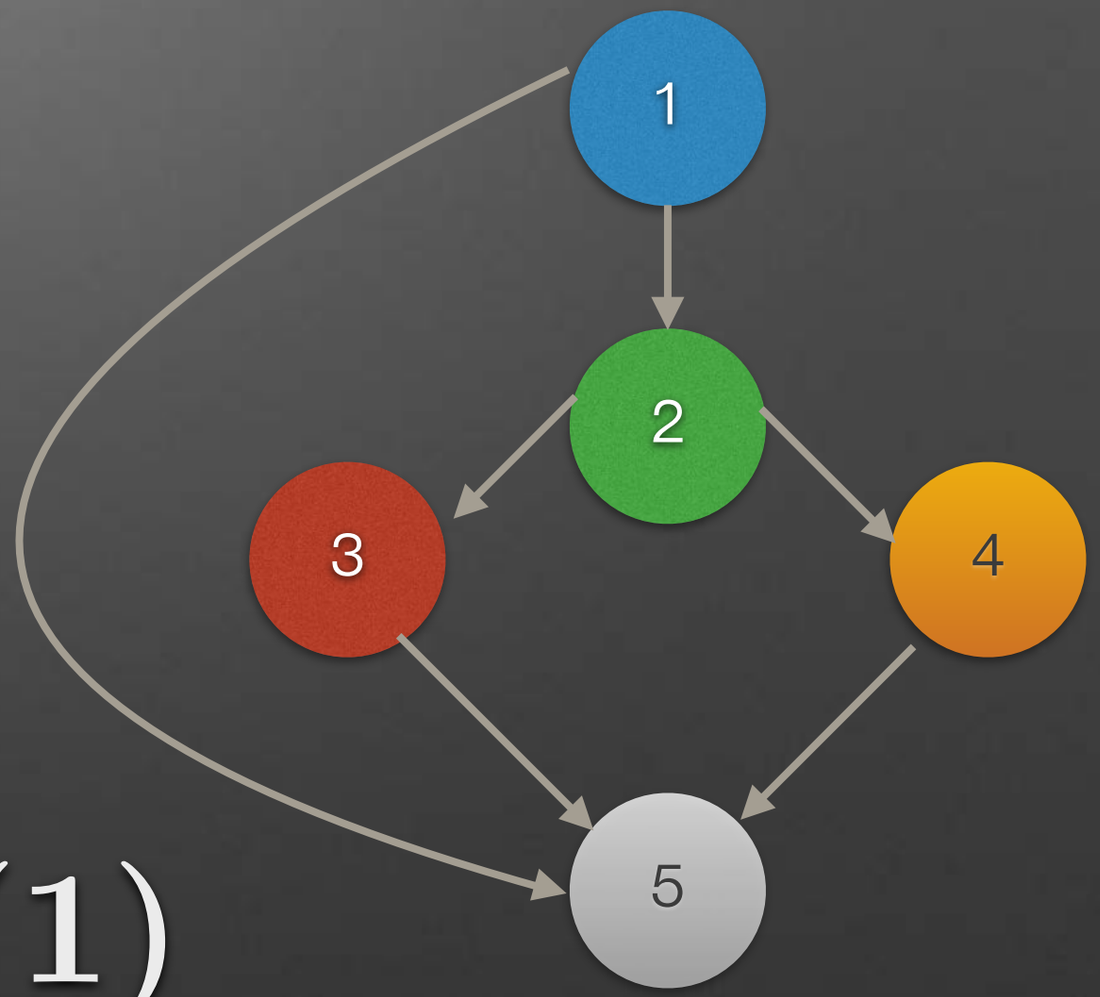
Connected Components

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

The Cyclomatic Complexity

If-Else Clause

```
insertion_procedure(int b, int c) {
(1) if (a == 111) {
(2)    if (b > c) {
(3)       a = b;
       } else {
(4)       a = c;
       }
    }
(5) Console.WriteLine(a);
}
```



$$M = 6 - 5 + 2(1)$$
$$= 3$$

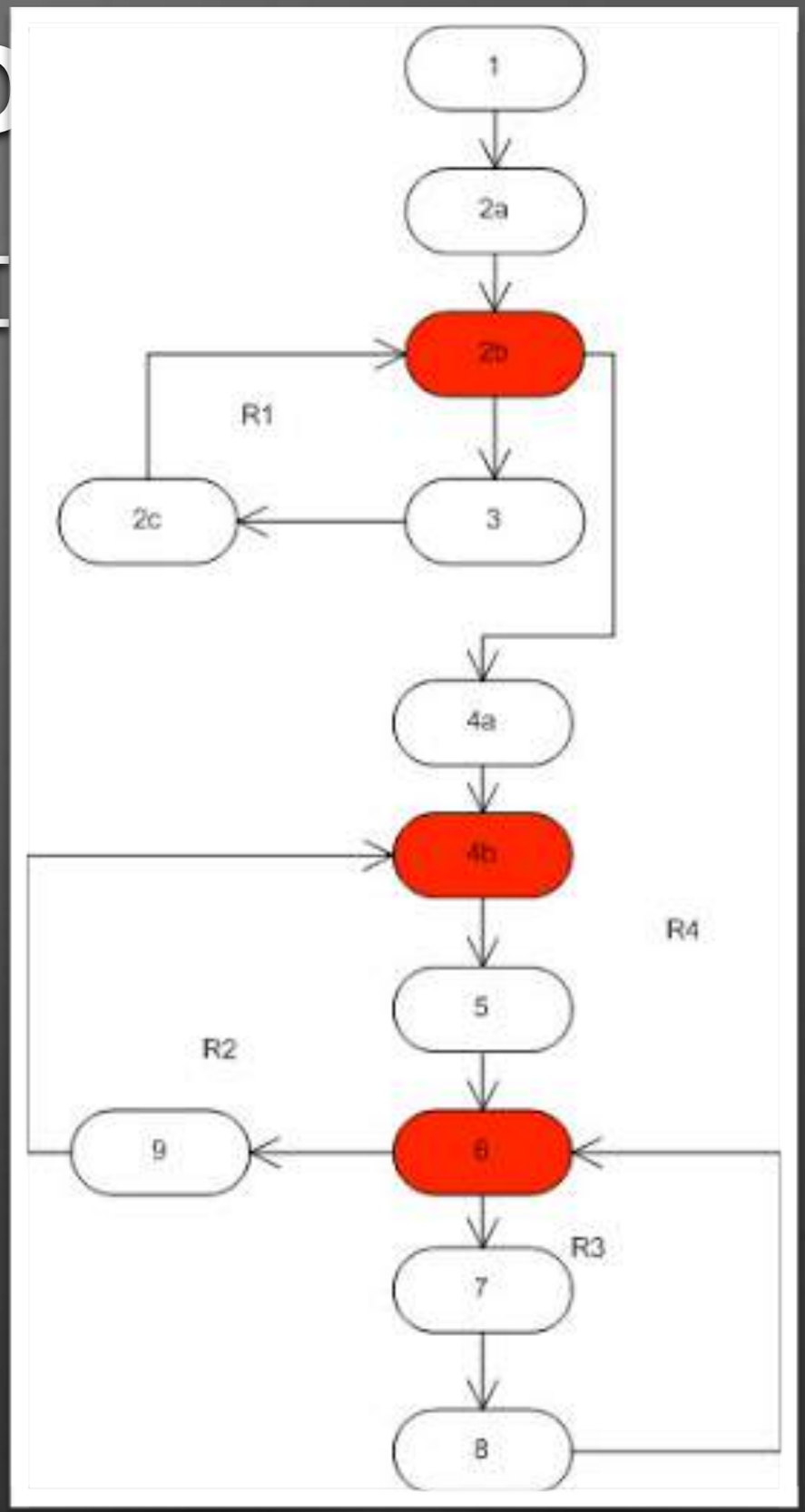
For/While Loops

```
insertion_procedure(int a[], int p [], int N) {
    int i, j, k;
    for (i = 0; i <= N; i ++) {
      p[i] = i;
    }
    for (i = 2; i <= N; i ++) {
      k = p[i];
      j = 1;
      while (a[p[j - 1]] > a[k]) {
        p[j] = p[j - 1];
        j --;
      }
      p[j] = k;
    }
}
```

http://stackoverflow.com/questions/2669977/control-flow-graph-cyclometric-complexity-for-folowing-procedure

For/While Lo

```
insertion_procedure(int a[], int p [
    int i, j, k;
    for (i = 0; i <= N; i ++) {
       p[i] = i;
    }
    for (i = 2; i <= N; i ++) {
       k = p[i];
       j = 1;
       while (a[p[j - 1]] > a[k]) {
          p[j] = p[j - 1];
          j --;
       }
       p[j] = k;
    }
}
```



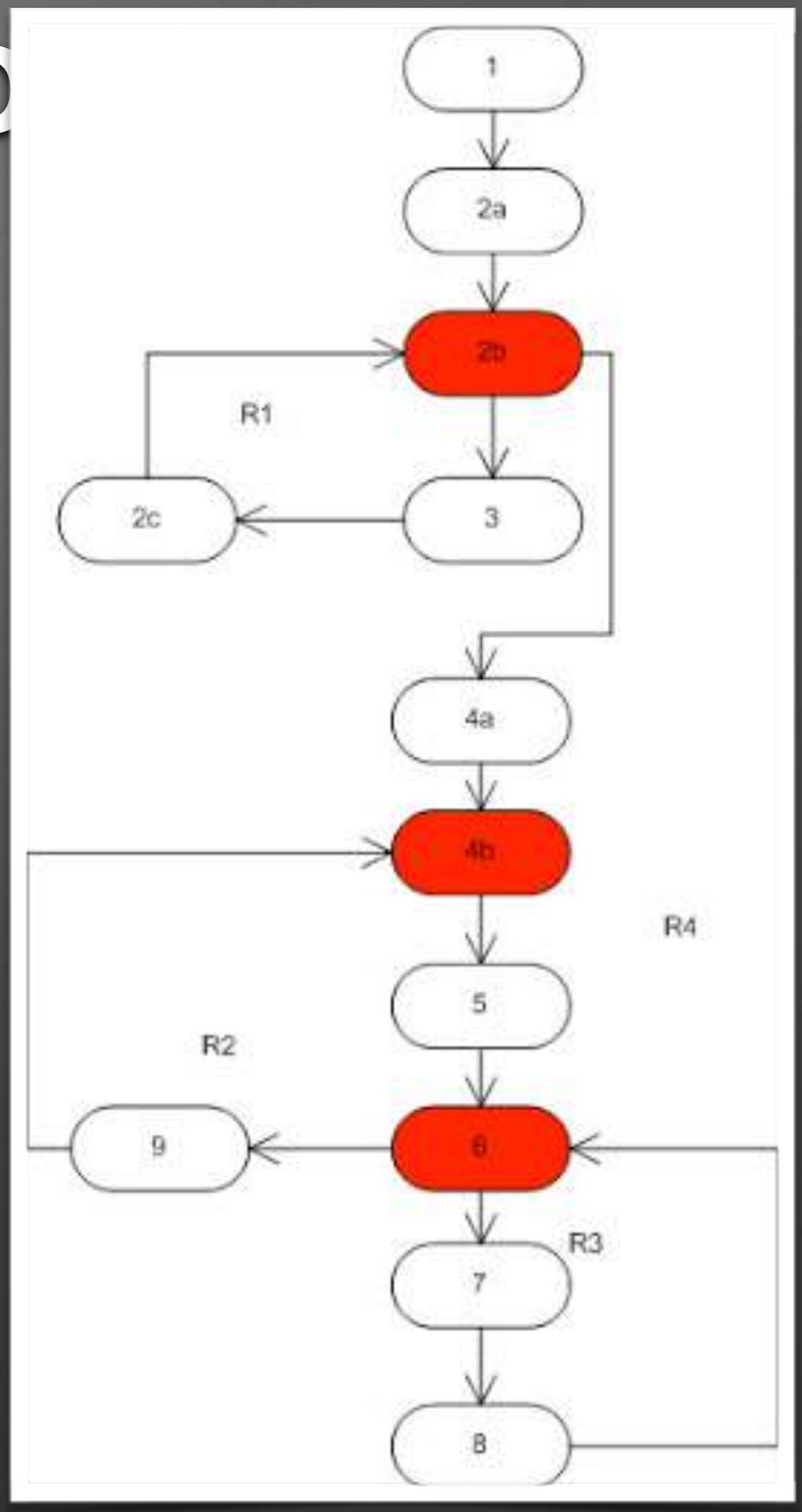
http://stackoverflow.com/questions/2669977/control-flow-graph-cyclometric-complexity-for-folowing-procedure

For/While Lo

14 edges

12 nodes

1 connected component



http://stackoverflow.com/questions/2669977/control-flow-graph-cyclometric-complexity-for-folowing-procedure

For/While Lo

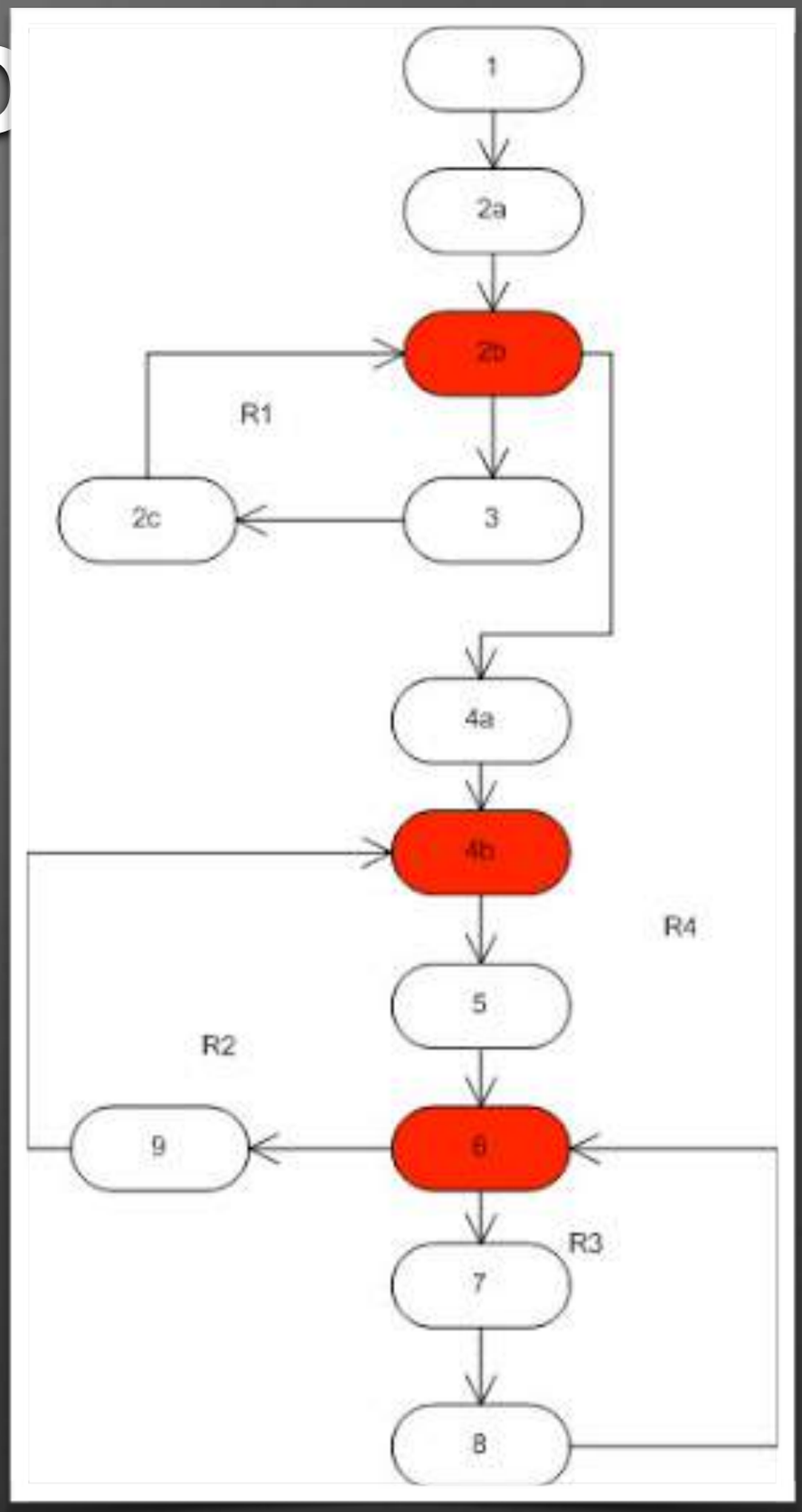
14 edges

12 nodes

1 connected component

$$M = 14 - 12 + 2(1)$$

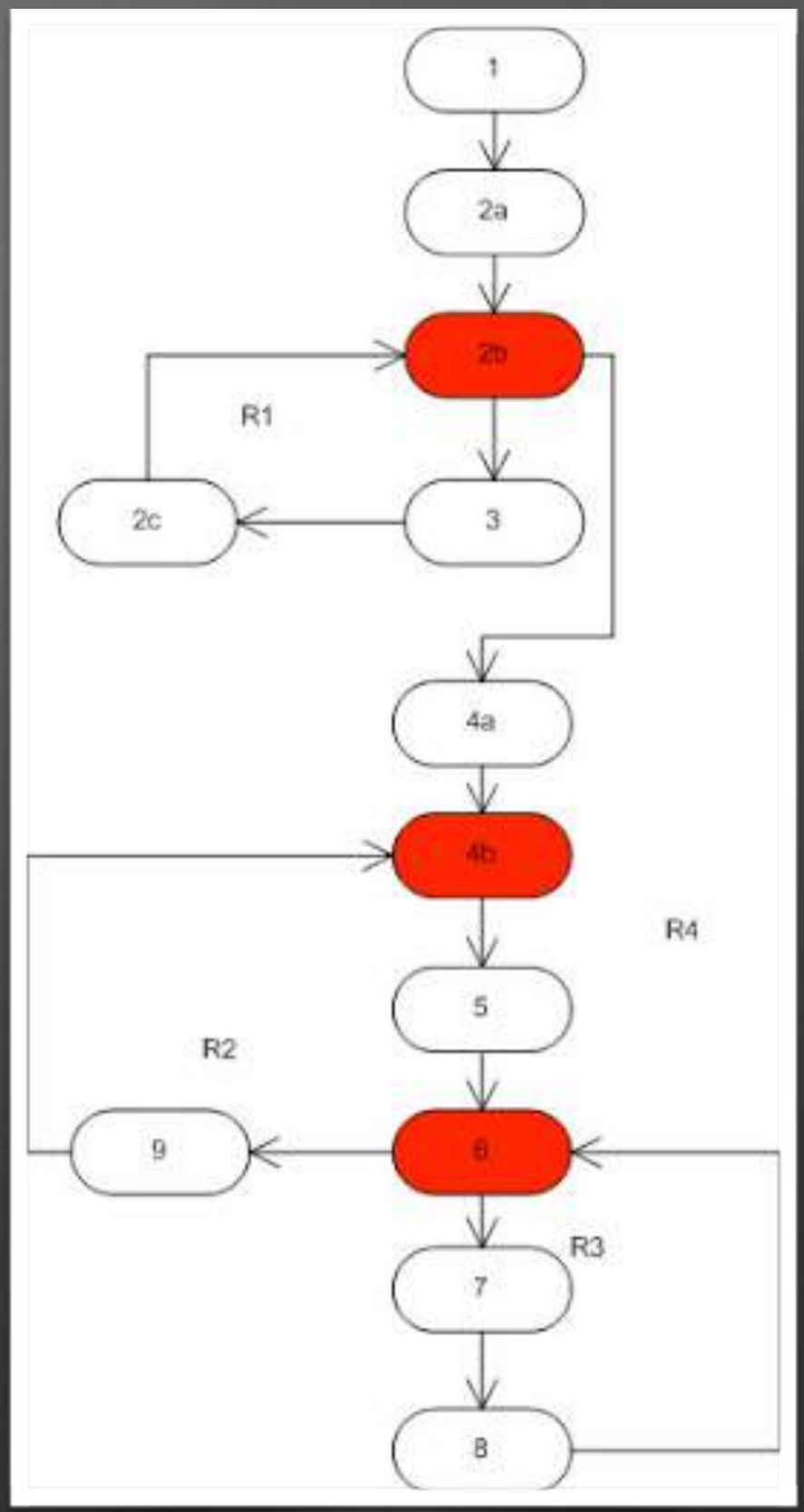
$$= 4$$



http://stackoverflow.com/questions/2669977/control-flow-graph-cyclometric-complexity-for-folowing-procedure

Alternative Formula

$$M = D + 1$$

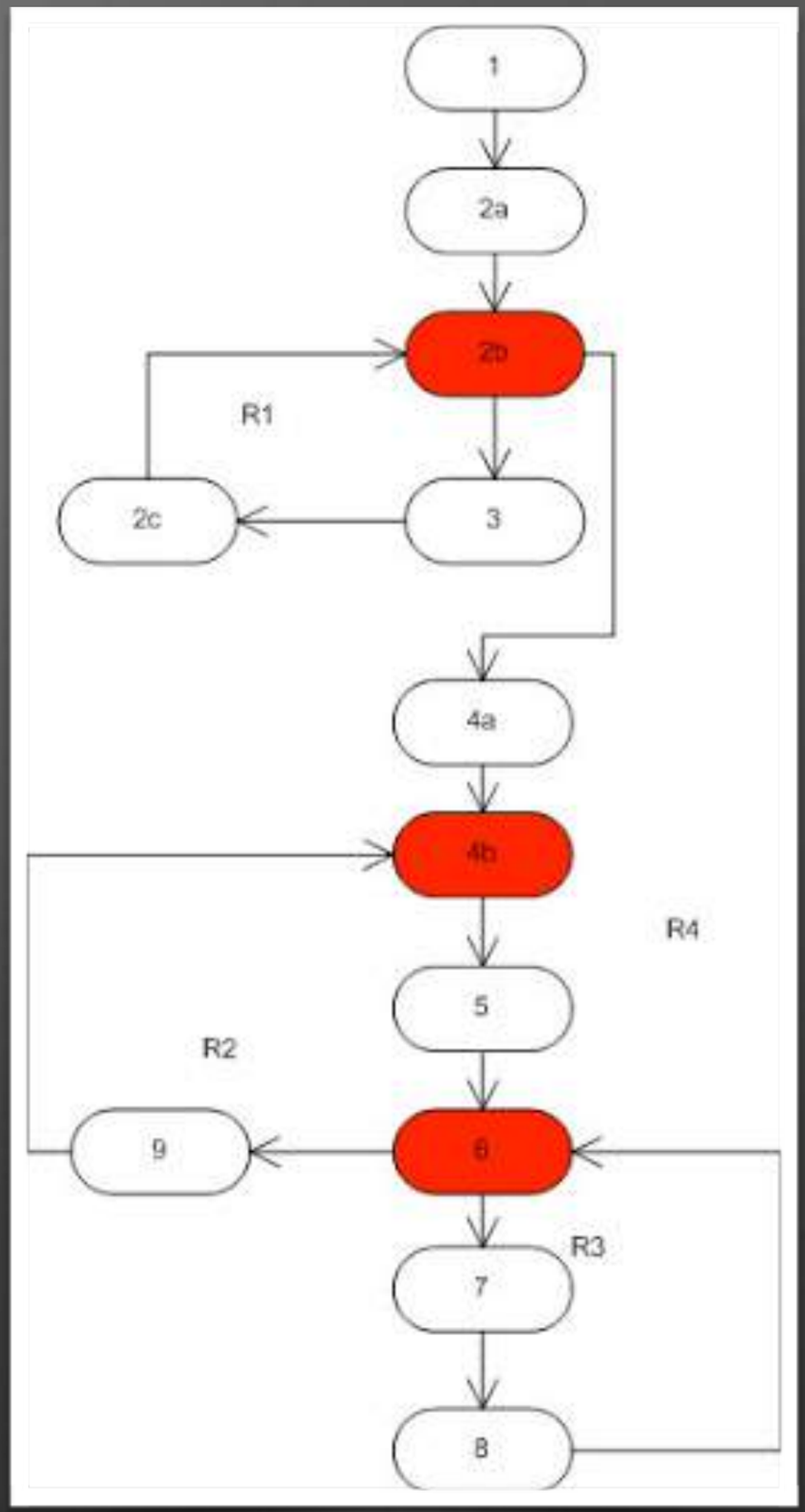


http://stackoverflow.com/questions/2669977/control-flow-graph-cyclometric-complexity-for-folowing-procedure

Alternative Formula

$$M = D + 1$$

No.Decision Points

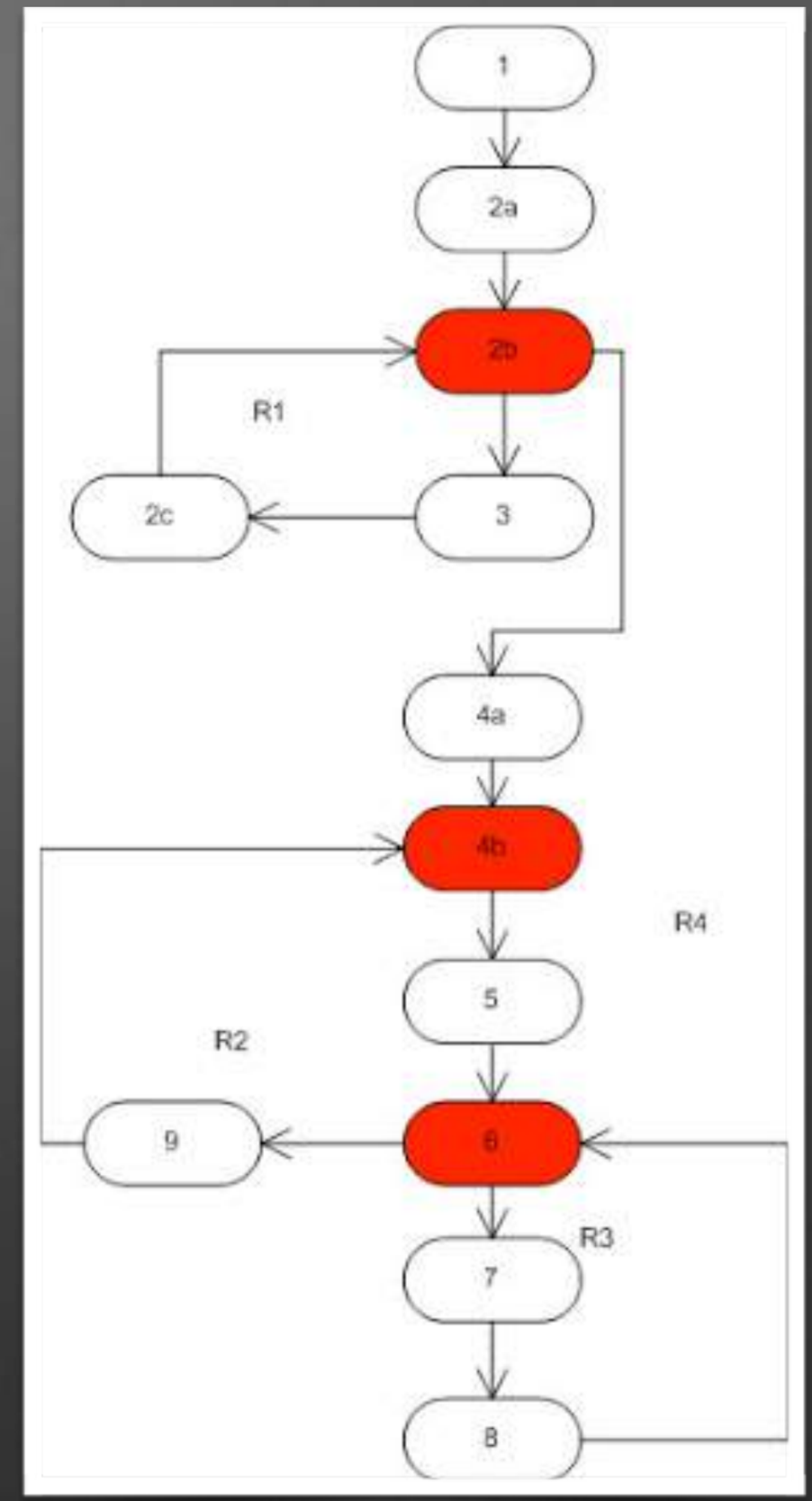


http://stackoverflow.com/questions/2669977/control-flow-graph-cyclometric-complexity-for-folowing-procedure

Alternative Formula

$$M = 3 + 1$$

$$= 4$$



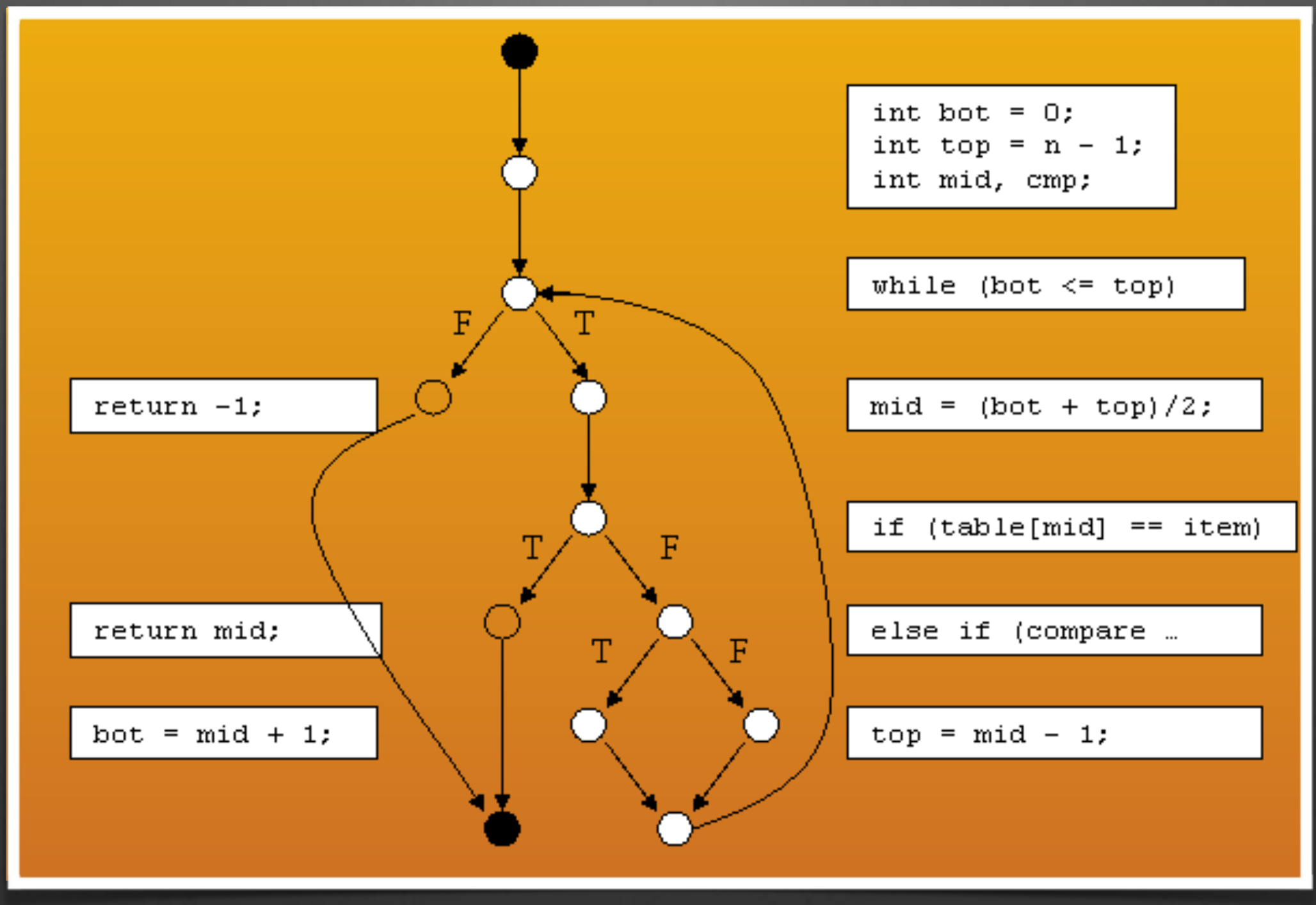
http://stackoverflow.com/questions/2669977/control-flow-graph-cyclometric-complexity-for-folowing-procedure

Combined Example

```
int BinSearch (char *item, char *table[], int n) {
    int bot = 0;
    int top = n - 1;
    int mid, cmp;
    while (bot <= top) {
      mid = (bot + top) / 2;
      if (table[mid] == item)
        return mid;
      else if (compare(table[mid], item) < 0)
        top = mid - 1;
      else
        bot = mid + 1;
    }
    return -1; // not found
}
```

http://www.whiteboxtest.com/cyclomatic-complexity.php

Combined Example



http://www.whiteboxtest.com/cyclomatic-complexity.php

Let’s Play a Game!

Other Metrics

Lines of Code

Test / Production Code Ratio

ABC Metrics

Branches

ABC Metrics

Assignments

Conditionals

Halstead Volume

For a given problem, Let:

- η_1 = the number of distinct operators
- η_2 = the number of distinct operands
- N_1 = the total number of operators
- N_2 = the total number of operands

From these numbers, several measures can be calculated:

- Program vocabulary: $\eta = \eta_1 + \eta_2$
- Program length: $N = N_1 + N_2$
- Calculated program length: $\hat{N} = \eta_1 \log_2 \eta_1 + \eta_2 \log_2 \eta_2$
- Volume: $V = N \times \log_2 \eta$
- Difficulty : $D = \frac{\eta_1}{2} \times \frac{N_2}{\eta_2}$
- Effort: $E = D \times V$

https://en.wikipedia.org/wiki/Halstead_complexity_measures

Maintainability Index

$$0 \leq \text{Maintainability Index} \leq 100$$

$$\text{Maintainability Index} = \max\left\{ \frac{(171 - 5.2 \times \ln \text{Halstead Volume} - 0.23 \times \text{Cyclomatic Complexity} - 16.2 \times \ln \text{Lines of Code}) \times 100}{171}, 0 \right\}$$

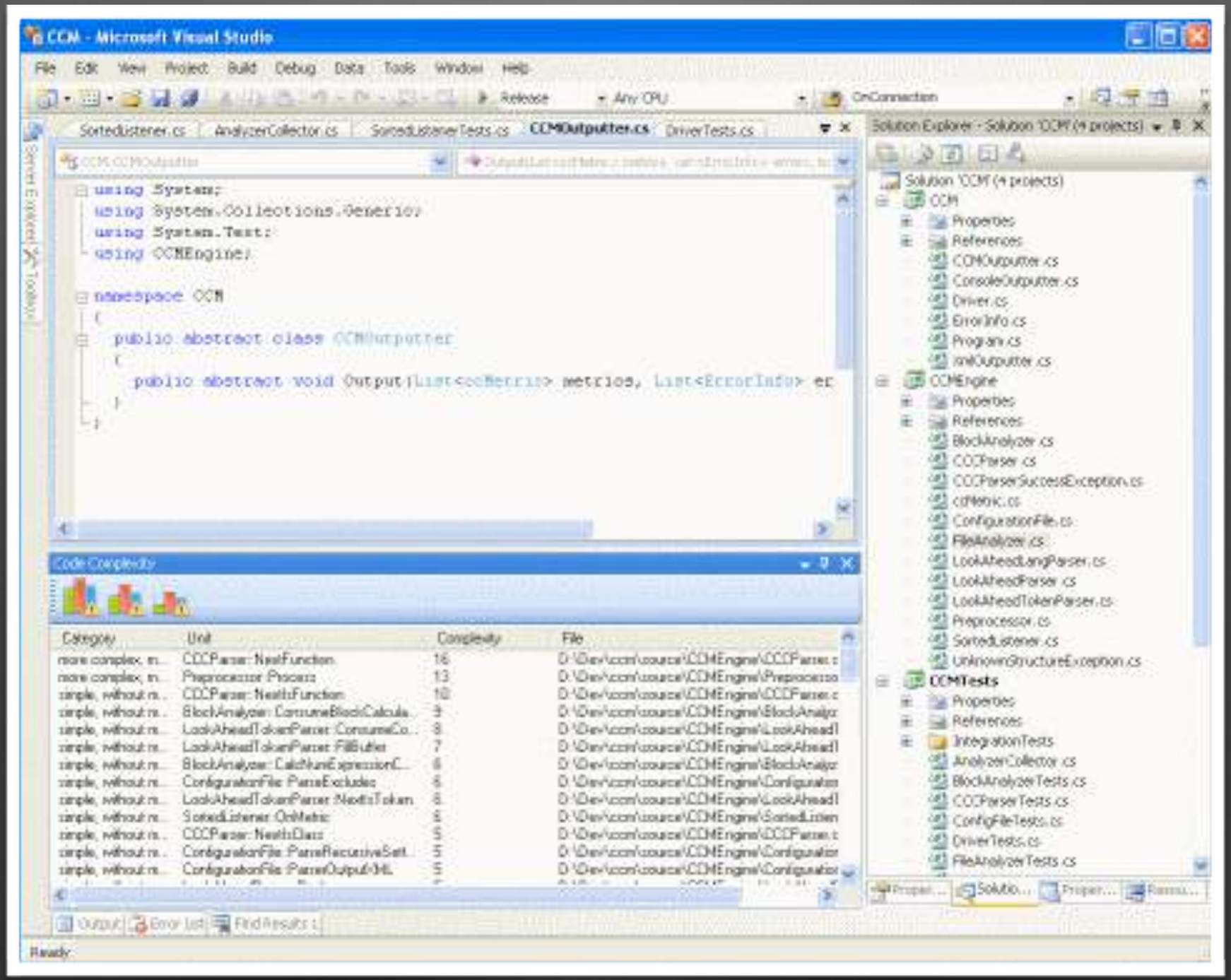
http://blogs.msdn.com/b/codeanalysis/archive/2007/11/20/maintainability-index-range-and-meaning.aspx

Code Duplication

```
var amount = parseFloat(userInput);
if(amount == 20) {
    banknotes_20 += 1;
    document.getElementById("output").value = "Accepted banknote";
}
else if(amount == 50) {
    banknotes_50 += 1;
    document.getElementById("output").value = "Accepted banknote";
}
else if(amount == 100) {
    banknotes_100 += 1;
    document.getElementById("output").value = "Accepted banknote";
}
else if(amount == 500) {
    banknotes_500 += 1;
    document.getElementById("output").value = "Accepted banknote";
}
else if(amount == 1000) {
    banknotes_1000 += 1;
    document.getElementById("output").value = "Accepted banknote";
}
```

Tools

CCM



http://www.blunck.info/ccm.html

Istanbul

Coverage Analysis ensures systematic exercise of the parser

Note: This is not a live (in-browser) code coverage report. The analysis is offline (../doc/index.html#coverage) (using Istanbul (https://github.com/yahoo/istanbul)).

Code coverage report for **esprima/esprima.js**

Statements: **99.72%** (1760 / 1765) Branches: **98.49%** (1107 / 1124) Functions: **100%** (154 / 154) Lines: **99.72%** (1760 / 1765)

```
 1       /*
 2         Copyright (C) 2012 Ariya Hidayat <ariya.hidayat@gmail.com>
 3         Copyright (C) 2012 Mathias Bynens <mathias@qiwi.be>
 4         Copyright (C) 2012 Joost-Wim Boekesteijn <joost-wim@boekesteijn.nl>
 5         Copyright (C) 2012 Kris Kowal <kris.kowal@cixar.com>
 6         Copyright (C) 2012 Yusuke Suzuki <utatane.tea@gmail.com>
 7         Copyright (C) 2012 Arpad Borsos <arpad.borsos@googlemail.com>
 8         Copyright (C) 2011 Ariya Hidayat <ariya.hidayat@gmail.com>
 9
10         Redistribution and use in source and binary forms, with or without
11         modification, are permitted provided that the following conditions are met:
12
13           * Redistributions of source code must retain the above copyright
14             notice, this list of conditions and the following disclaimer.
15           * Redistributions in binary form must reproduce the above copyright
16             notice, this list of conditions and the following disclaimer in the
17             documentation and/or other materials provided with the distribution.
18
19         THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
20         AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
21         IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
22         ARE DISCLAIMED. IN NO EVENT SHALL <COPYRIGHT HOLDER> BE LIABLE FOR ANY
23         DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
24         (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
25         LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
26         ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
27         (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
28         THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
29       */
30
31       /*jslint bitwise:true plusplus:true */
32       /*global esprima:true, define:true, exports:true, window: true,
33       throwError: true, generateStatement: true, peek: true,
34       parseAssignmentExpression: true, parseBlock: true, parseExpression: true,
35       parseFunctionDeclaration: true, parseFunctionExpression: true,
36       parseFunctionSourceElements: true, parseVariableIdentifier: true,
37       parseLeftHandSideExpression: true,
38       parseStatement: true, parseSourceElement: true */
39
40     1 (function (root, factory) {
41     1     'use strict';
42
43           // Universal Module Definition (UMD) to support AMD, CommonJS/Node.js,
44           // Rhino, and plain browser loading.
45     1     I if (typeof define === 'function' && define.amd) {
46               define(['exports'], factory);
47     1     } else E if (typeof exports !== 'undefined') {
48     1         factory(exports);
```

https://github.com/yahoo/istanbul

Plato

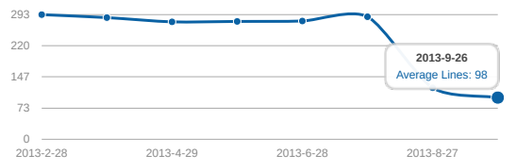
Summary

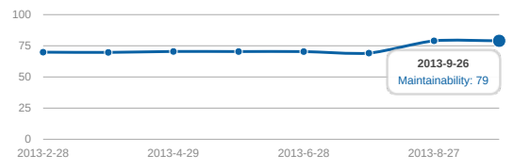
Total/Average Lines ℹ

7660 / 98

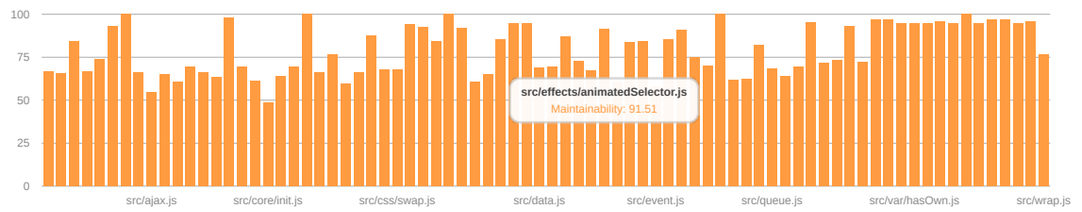
Average Maintainability ℹ (http://blogs.msdn.com/b/codeanalysis/archive/2007/11/20/maintainability-index-range-and-meaning.aspx)

79.00

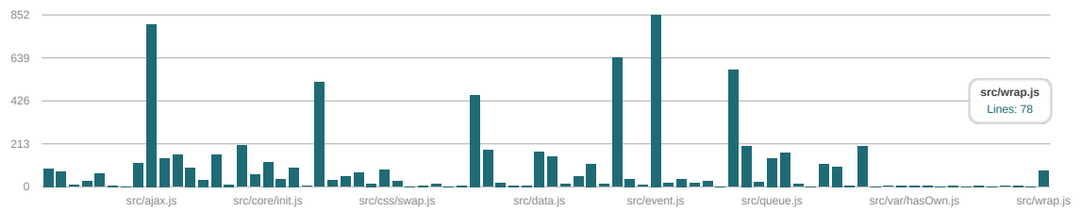




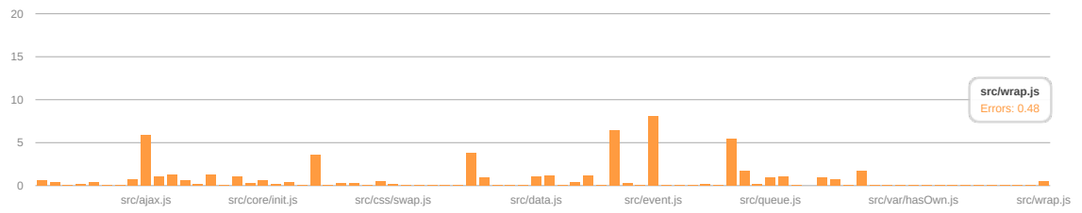
Maintainability ℹ (http://blogs.msdn.com/b/codeanalysis/archive/2007/11/20/maintainability-index-range-and-meaning.aspx)



Lines of code ℹ



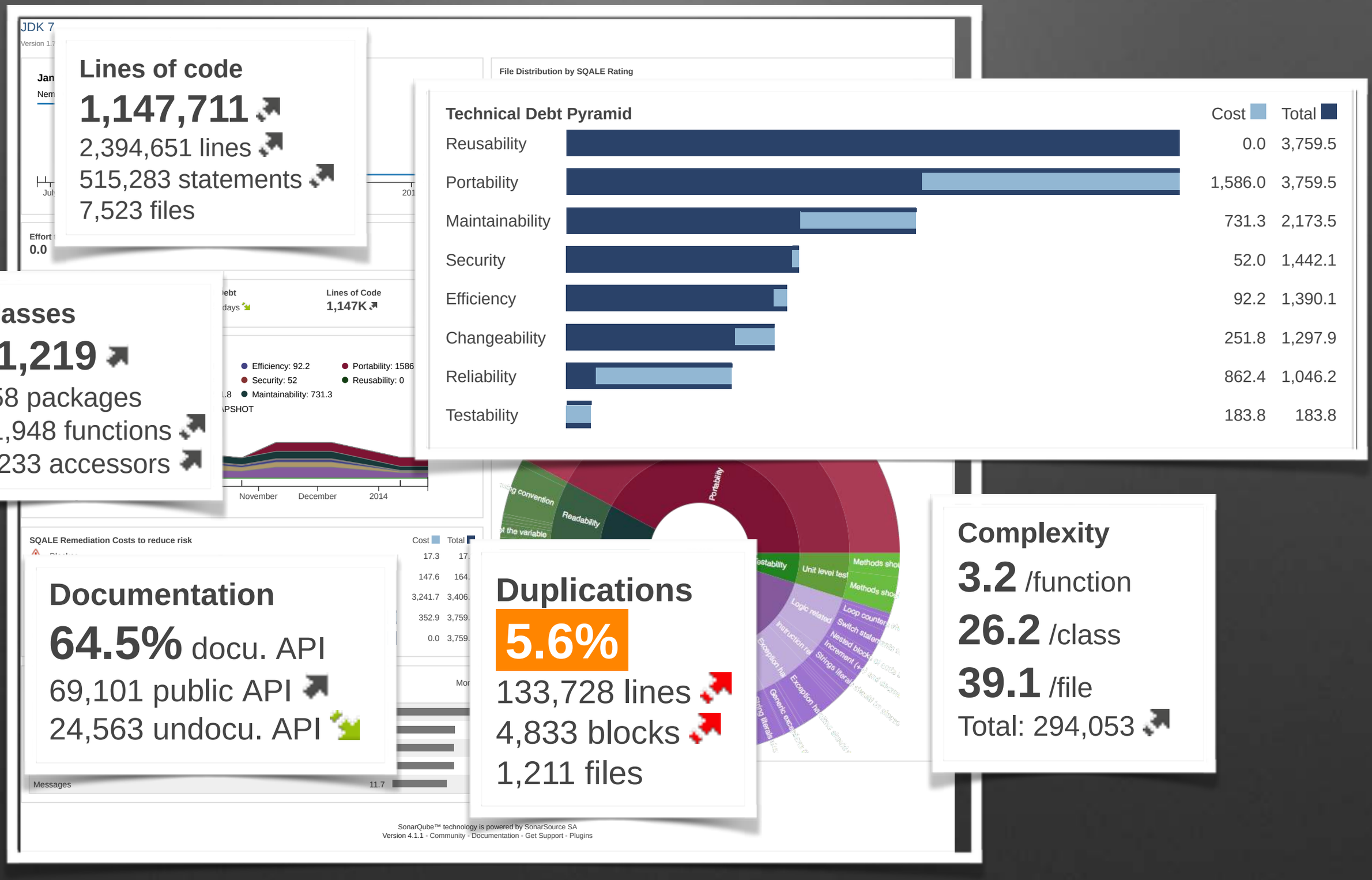
Estimated errors in implementation ℹ (http://en.wikipedia.org/wiki/Halstead_complexity_measures)



https://github.com/es-analysis/plato

SonarQube

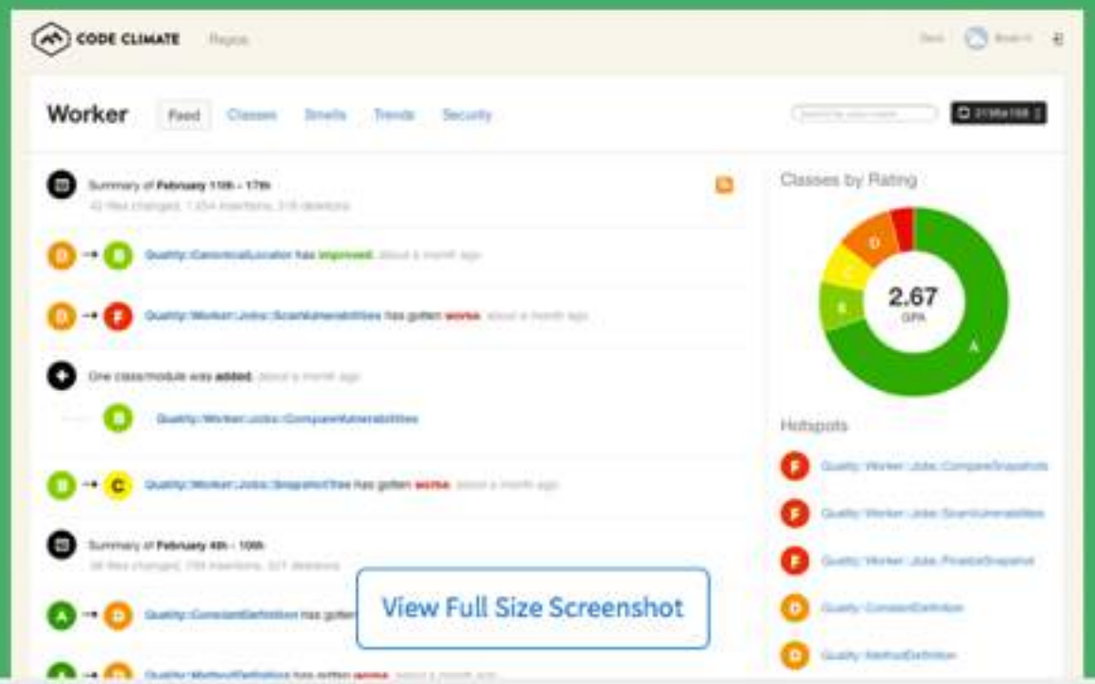
http://www.sonarqube.org/



Code Climate

Automated Code Review

Quality & security analysis for Ruby on Rails and Javascript.



- **Raise the Visibility of Code Quality**
  Increase front-end and back-end awareness within your team.
- **Get Immediate Feedback**
  Address code smells before they become technical debt.
- **Keep Your App and Data Safe**
  Fix risky code before it hits production.

Take control of technical debt today with Code Climate.
No Credit Card Required

Get Started – Free for 14 Days

http://codeclimate.com/

Summary

Avoid Technical Debts

Questions?