JUG Switzerland

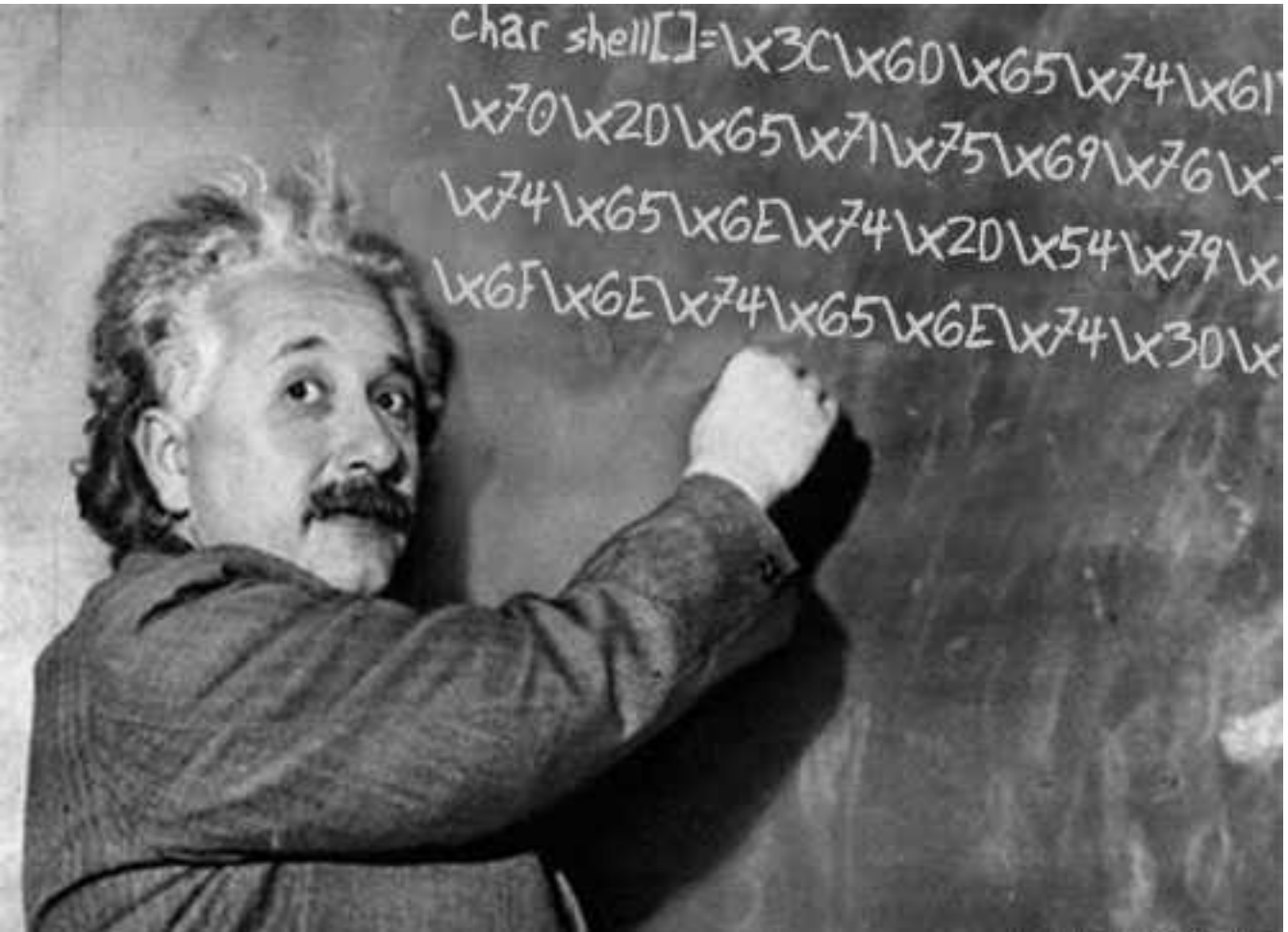
Continuous Inspection

Fight back the 7 deadly sins of the developer !

Olivier Gaudin
olivier.gaudin@sonarsource.com
@gaudol

Back in old times

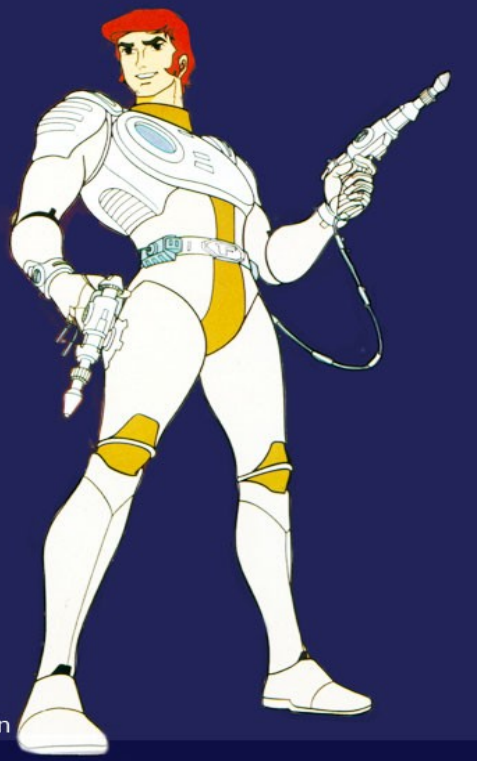
The genius


char shell[]=\x3C\x6D\x65\x74\x61
\x70\x20\x65\x71\x75\x69\x76\x
\x74\x65\x6E\x74\x20\x54\x79\x
\x6F\x6E\x74\x65\x6E\x74\x3D\x

The super hero

Au fin fond de l'Univers, à des années
et des années-lumière de la Terre,
Veille celui que le gouvernement
intersidéral appelle quand il n'est plus
Capable de trouver une solution à
ses problèmes, quand il ne reste plus
Aucun espoir :

le Capitaine FLAM !



Capitaine Flam © 1980 Tôei Animation

This is my toy

Fear of changes

Industrialisation has entered the game...

- Project under version control
- Project under continuous integration
- Technical and functional traceability

What is the mission of today's developer ?

?

Sustainable development

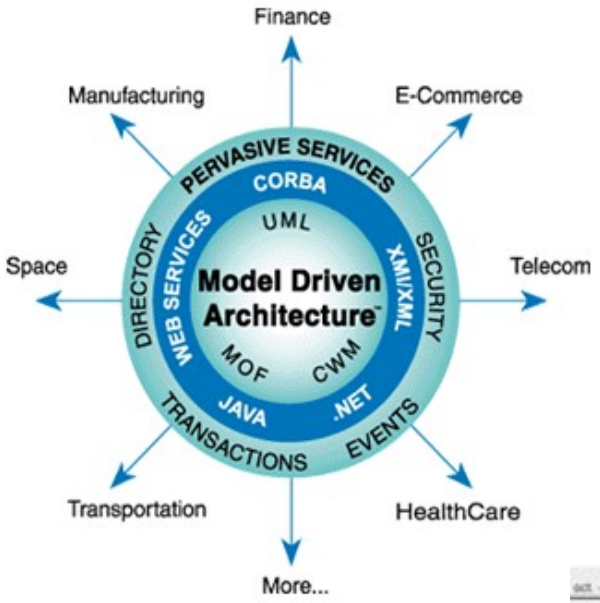
(Almost) Everything is maintenance !

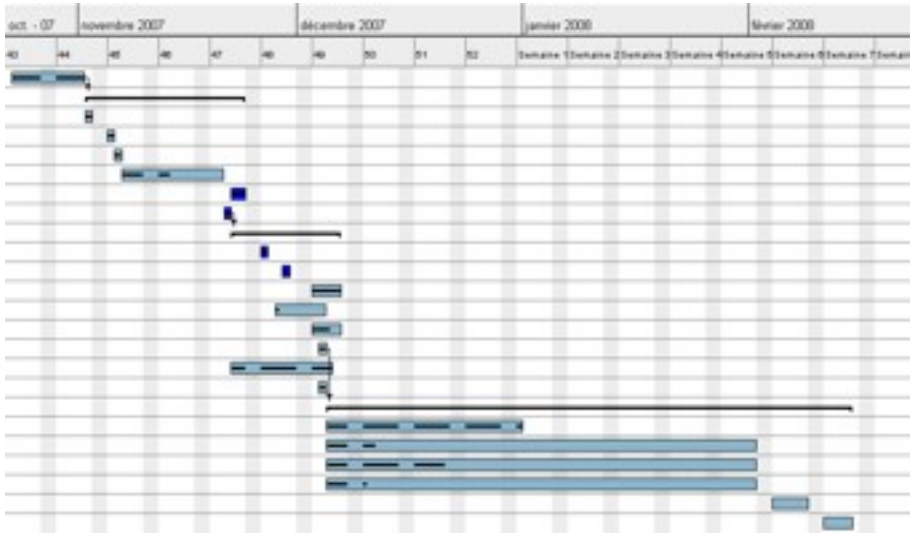
Creation of an application

Maintenance of an application

Nothing is more important than code

But source code is nothing alone





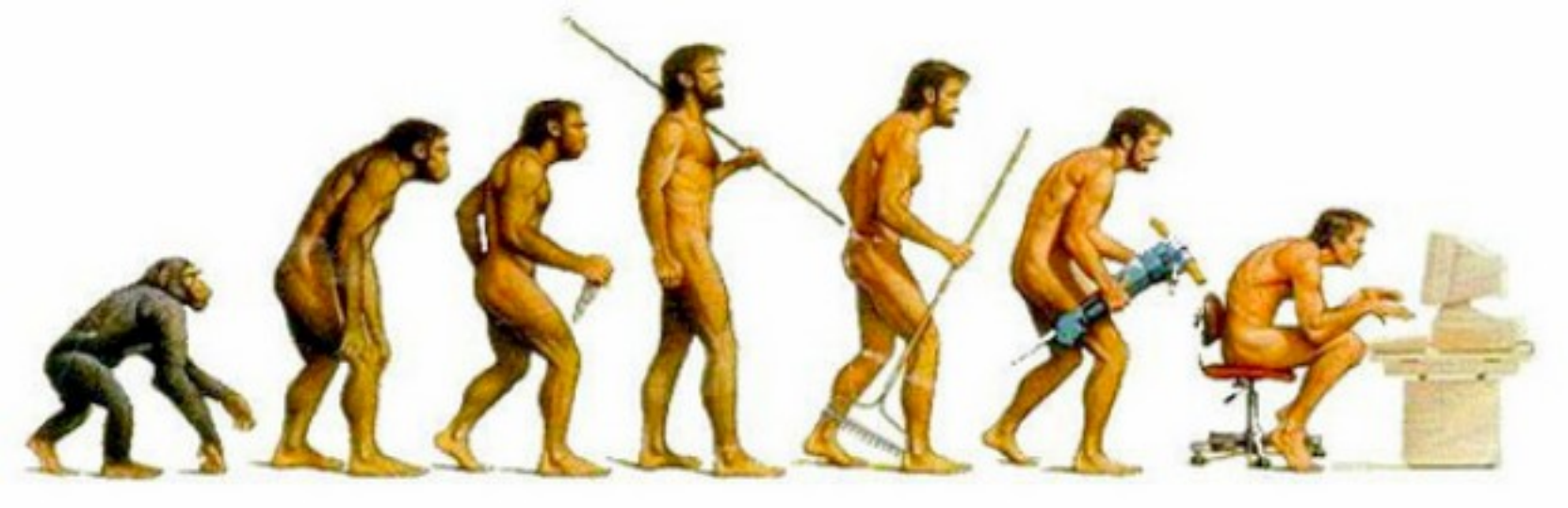
Old times are over

Developing for others

Methodology

Transparency

Software factories evolve



VI / Emacs

Makefile

SCM

Issue Tracker

Continuous Integration

IDE Refactoring

Unit Tests

Continuous Inspection

Pushed by requirements

Configuration Manager

- No change should be authorized to production system without being in configuration manager
- The complete version of an application should be found easily in the source control manager

Pushed by requirements

Continuous Integration

- Projects in SCM can be built by anybody at any time
- Executing unit tests is part of the build process
- The output of a build is an artifact “ready to be used”
- If one of those requirements is not fulfilled, nothing is more important than fixing it

Pushed by requirements

Continuous Inspection

- Any new code should ship with corresponding unit tests
- No new method should exceed a pre-defined level of complexity
- No cycle between packages should be introduced
- …

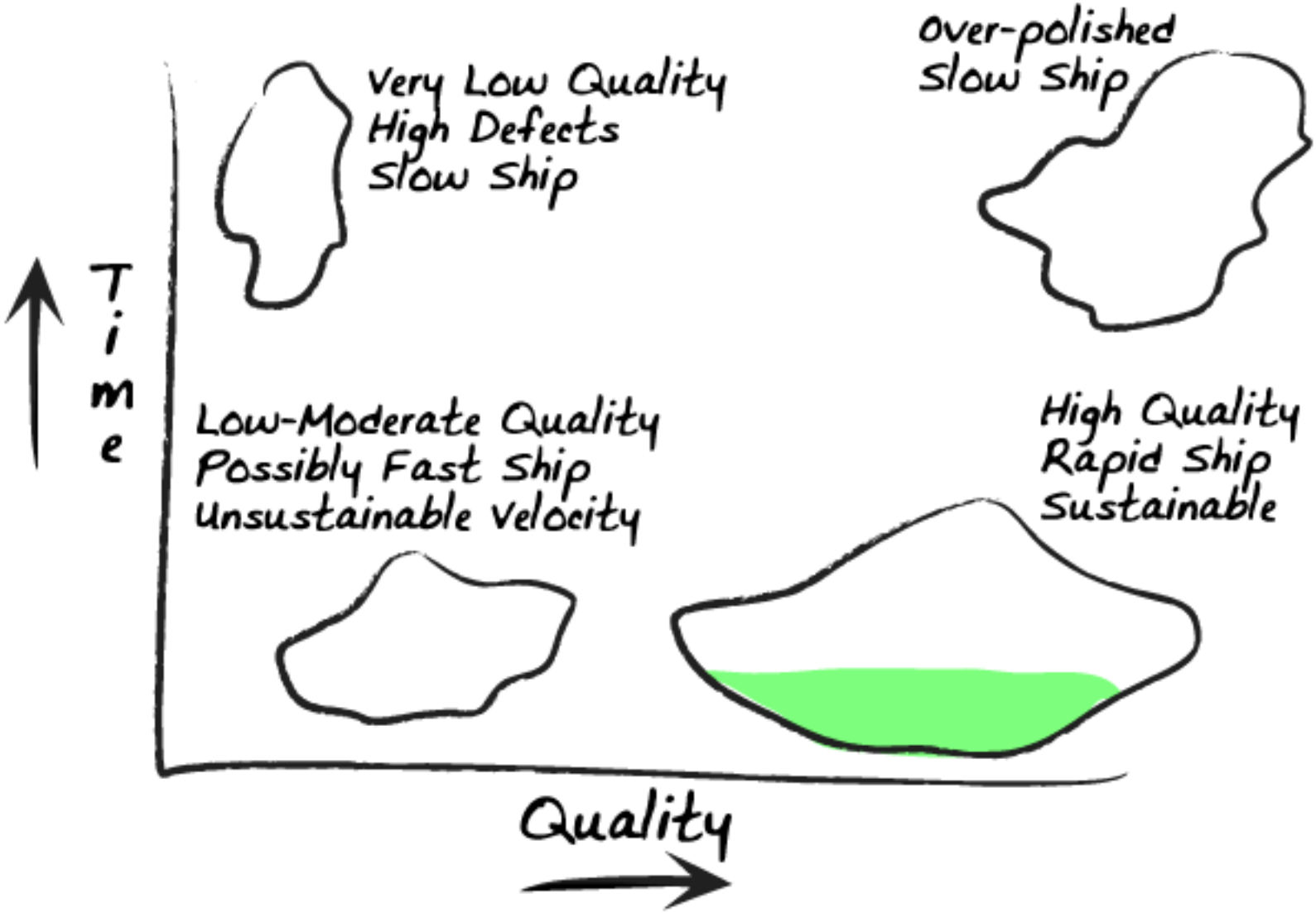
BUT...

Maturity steps should be followed

- Insuring technical traceability
  Configuration Manager
- Insuring functional traceability
  Issue Manager
- Insuring build stability
  Continuous Integration
- Insuring source code quality
  Continuous Inspection (Sonar)

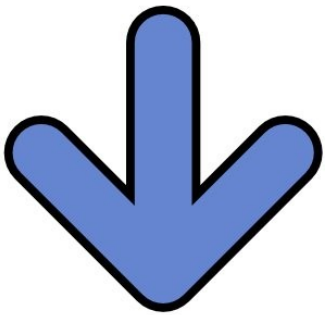
Quality versus Productivity

Extract from xprogramming.com



The end does not justify the means

*Doing the **right software***



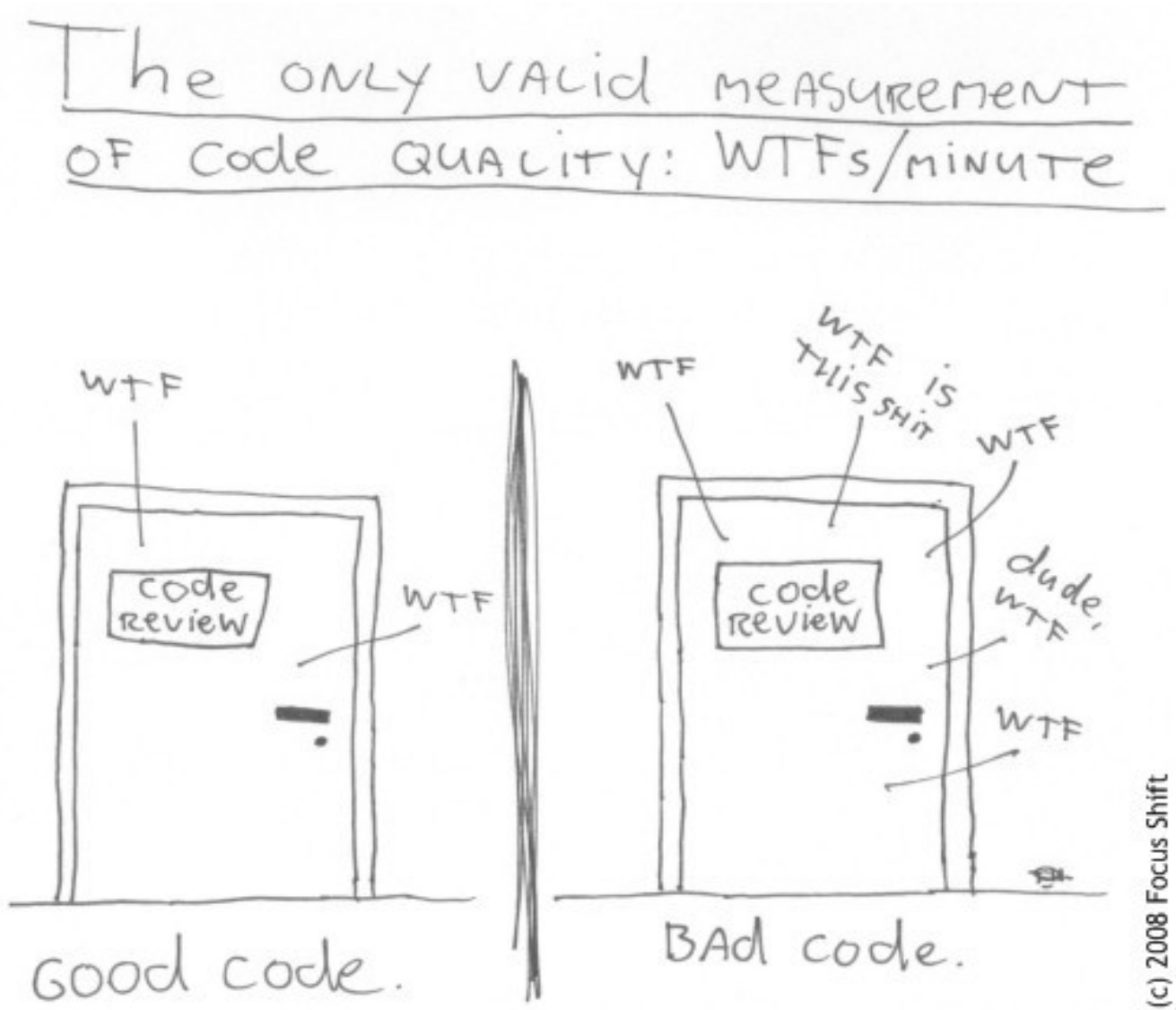
*Doing the **software right***

What is quality?

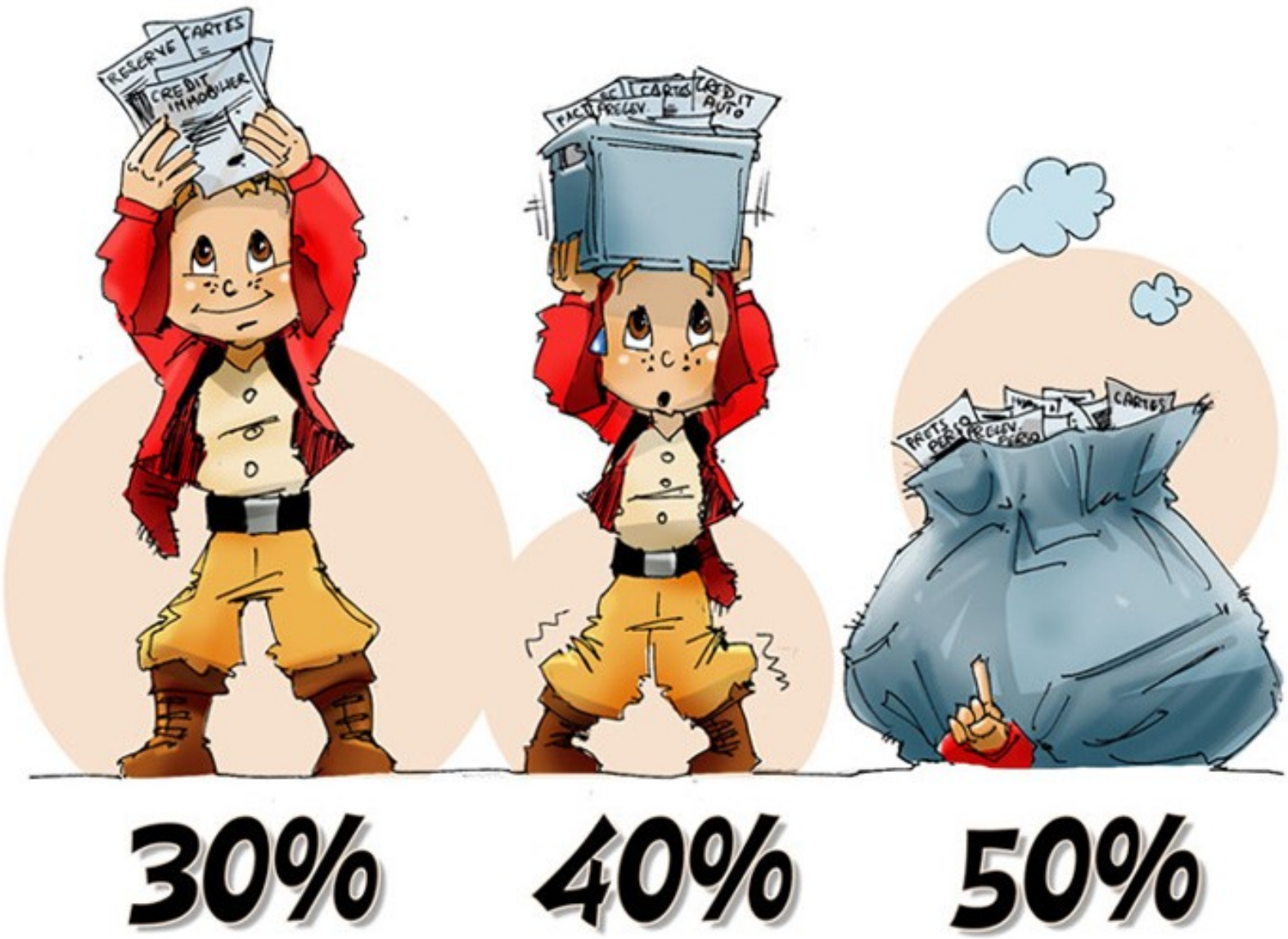
« A well-written program is a program
where the cost of implementing a feature
is constant throughout the program's lifetime. »

Itay Maman

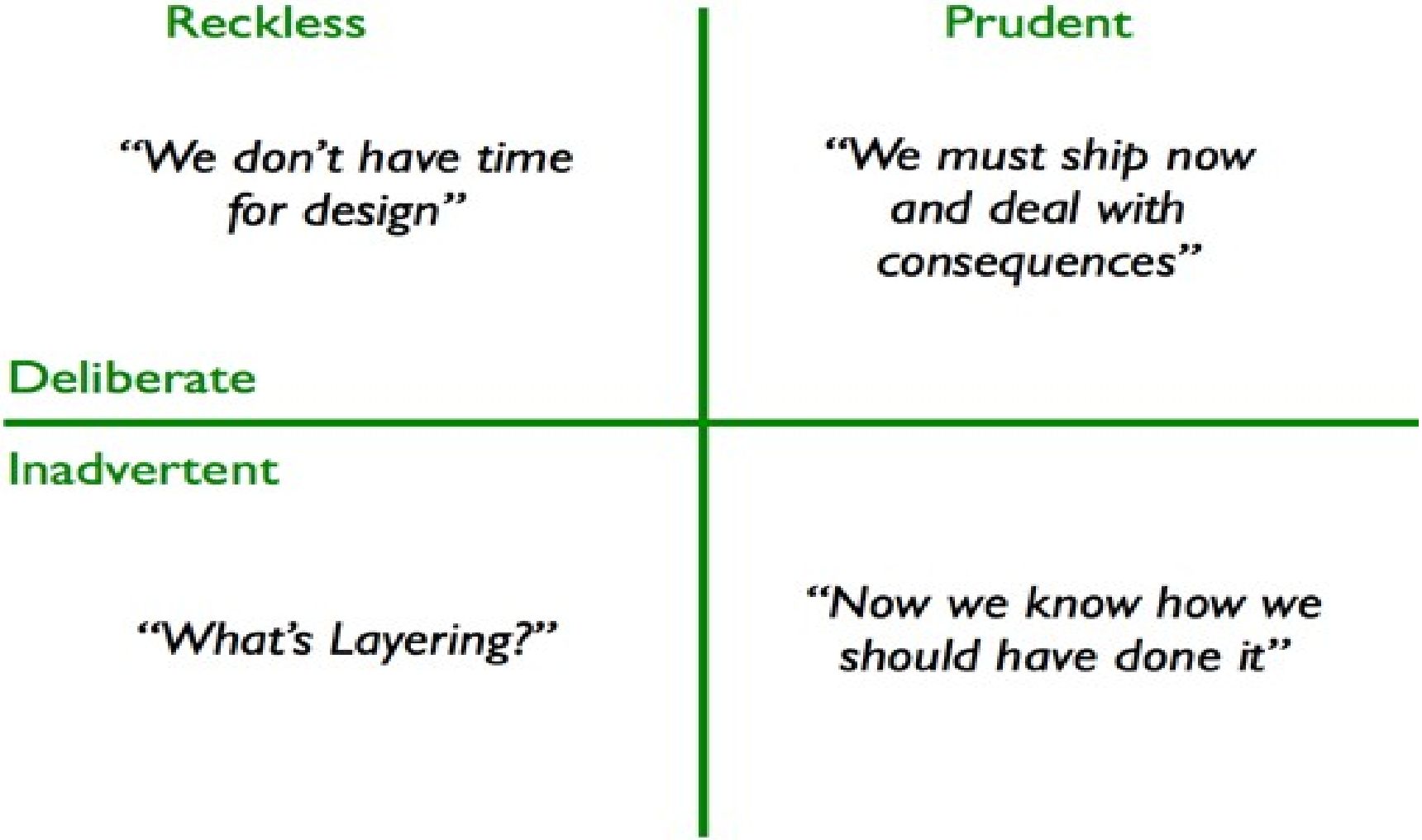
How to measure quality ?



The technical debt



The various types of Debt



Extract from http://martinfowler.com/bliki/TechnicalDebtQuadrant.html

The 7 deadly sins

of the developer

Sins

Technical Debt

The 7 deadly sins ?

Applied to source code

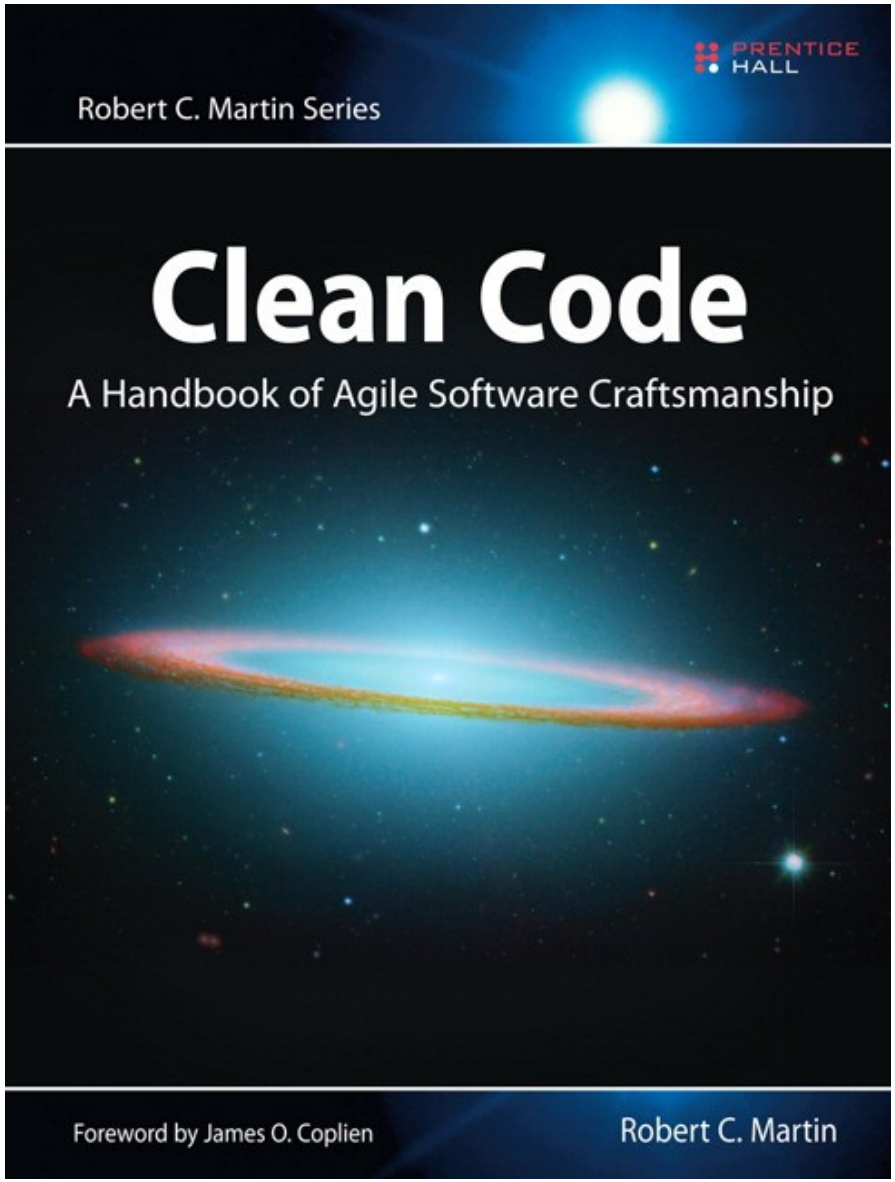
- ?
- ?
- ?
- ?
- ?
- ?
- ?

The 7 deadly sins ?

Applied to source code

- Duplications
- Bad distribution of complexity
- Spaghetti Design
- Lack of unit tests
- No coding standards
- Potential bugs
- Not enough or too many comments

To get back on track



Once and only once (Kent Beck)

Duplicated code is an opportunity to raise the level of abstraction and improve the design

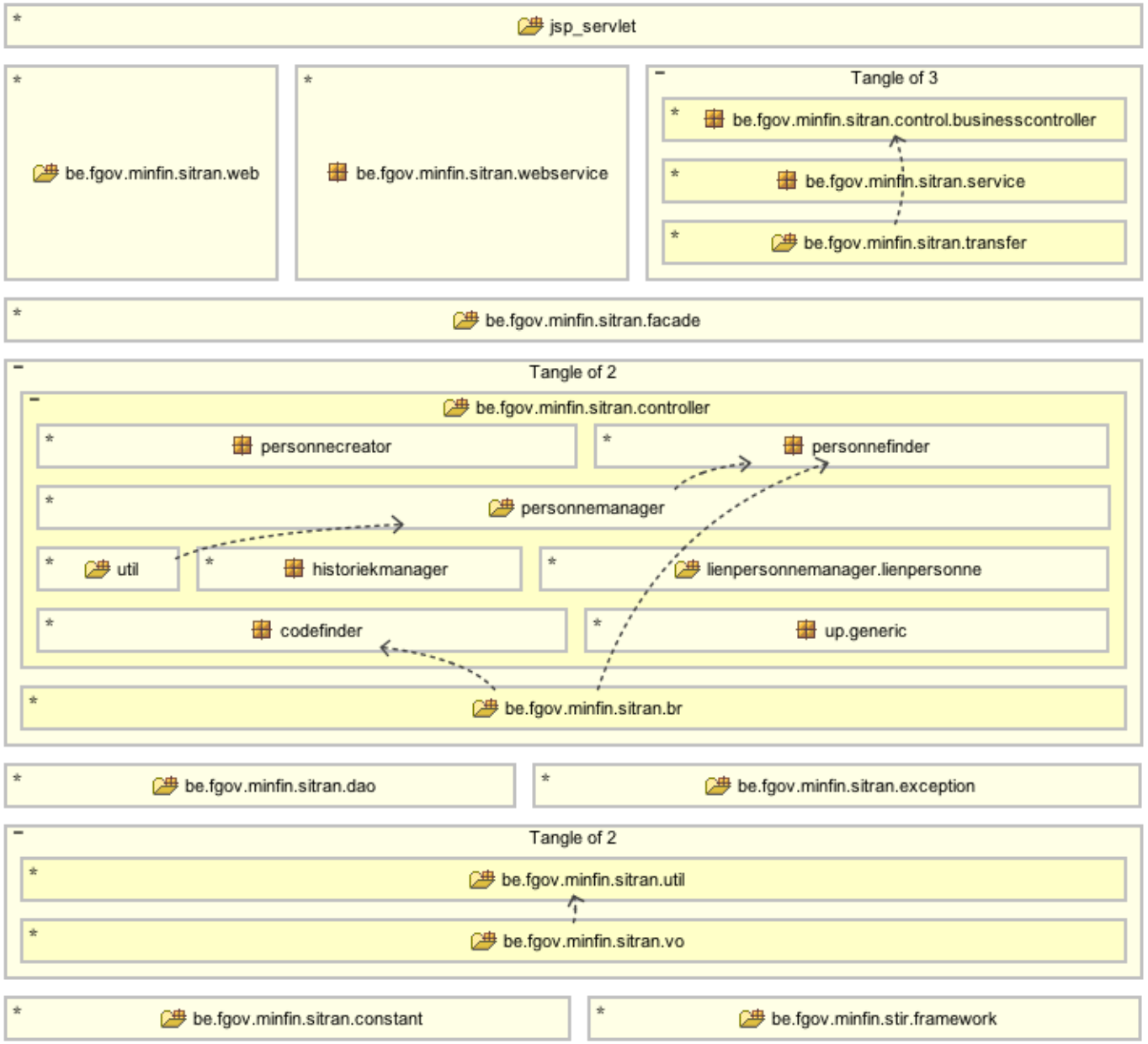
Bad distribution of complexity

- Do you choose :
  - 1 method with complexity of 30
  - 10 methods with complexity of 3

```
if (size > 0) {
    Object otherValue = null;
    switch (size) {  // drop through
        case 3:
            if (other.containsKey(key3) == false) {
                return false;
            }
            otherValue = other.get(key3);
            if (value3 == null ? otherValue != null : !value3.equals(otherValue)) {
                return false;
            }
        case 2:
            if (other.containsKey(key2) == false) {
                return false;
            }
            otherValue = other.get(key2);
            if (value2 == null ? otherValue != null : !value2.equals(otherValue)) {
                return false;
            }
```

Architecture layers

Cycles are plain as the nose on one's face



Insufficient unit tests

- Thank you for adding a new case without regression...

```
    public V put(K key, V value) {
0%      if (delegateMap != null) {
            return delegateMap.put(key, value);
        }
        // change existing mapping
0%      if (key == null) {
0%          switch (size) {  // drop through
                case 3:
0%                  if (key3 == null) {
                        V old = value3;
                        value3 = value;
                        return old;
                    }
                case 2:
0%                  if (key2 == null) {
                        V old = value2;
                        value2 = value;
                        return old;
                    }
                case 1:
0%                  if (key1 == null) {
                        V old = value1;
                        value1 = value;
                        return old;
                    }
            }
        } else {
0%          if (size > 0) {
                int hashCode = key.hashCode();
0%              switch (size) {  // drop through
                    case 3:
0%                      if (hash3 == hashCode && key.equals(key3)) {
                            V old = value3;
                            value3 = value;
                            return old;
                        }
                    case 2:
0%                      if (hash2 == hashCode && key.equals(key2)) {
                            V old = value2;
                            value2 = value;
                            return old;
                        }
                    case 1:
0%                      if (hash1 == hashCode && key.equals(key1)) {
                            V old = value1;
                            value1 = value;
                            return old;
                        }
                }
            }
        }
```

Coding standards


```
try {
   computeOrder(order);
} catch (RuntimeException e) {
   System.out.println("The order can't be computed!");
}
```

Potential bugs

```
if (listeners == null)
    listeners.remove(listener);
```

Sun java : JDK1.6.0, b105,
sun.awt.x11.XMSelection
lines 243-244

A comment must be useful

Or not exist

- To reinforce the logic

```
String listItemContent = match.group(3).trim();
// the trim is really important. It removes the starting
// spaces that could cause the item to be recognized
// as another list
```

- To add some dynamics

A typical invocation sequence is thus

```
Pattern p = Pattern.compile("a*b");
Matcher m = p.matcher("aaaaab");
boolean b = m.matches();
```

- Anti-patterns

```
i++; // increment i
```

```
// 04-Sep-2003 - Implemented Comparable. Updated the isInRange javadocs
// 05-Jan-2005 - Fixed ug in addYears() method
```

The mission of Sonar

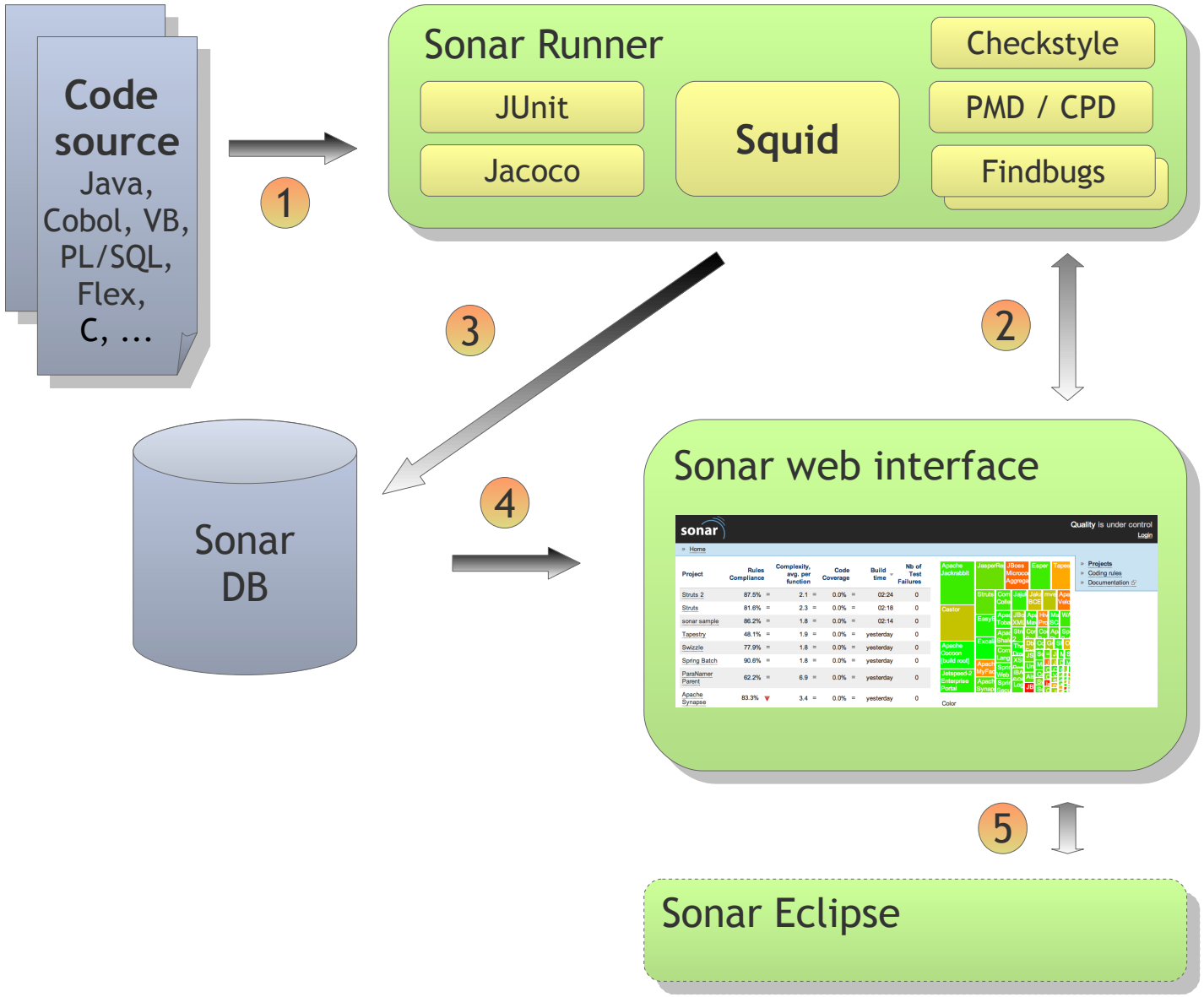
Declare **open** the **hunt** of the developer's **7 deadly sins**

The mission of Sonar

More seriously

Augment everybody's **capability** to reduce, reuse and recycle source code

The heart of Sonar



1- `mvn sonar:sonar`
or
`ant sonar`
or
`sonar-runner`

2- `http://sonar`

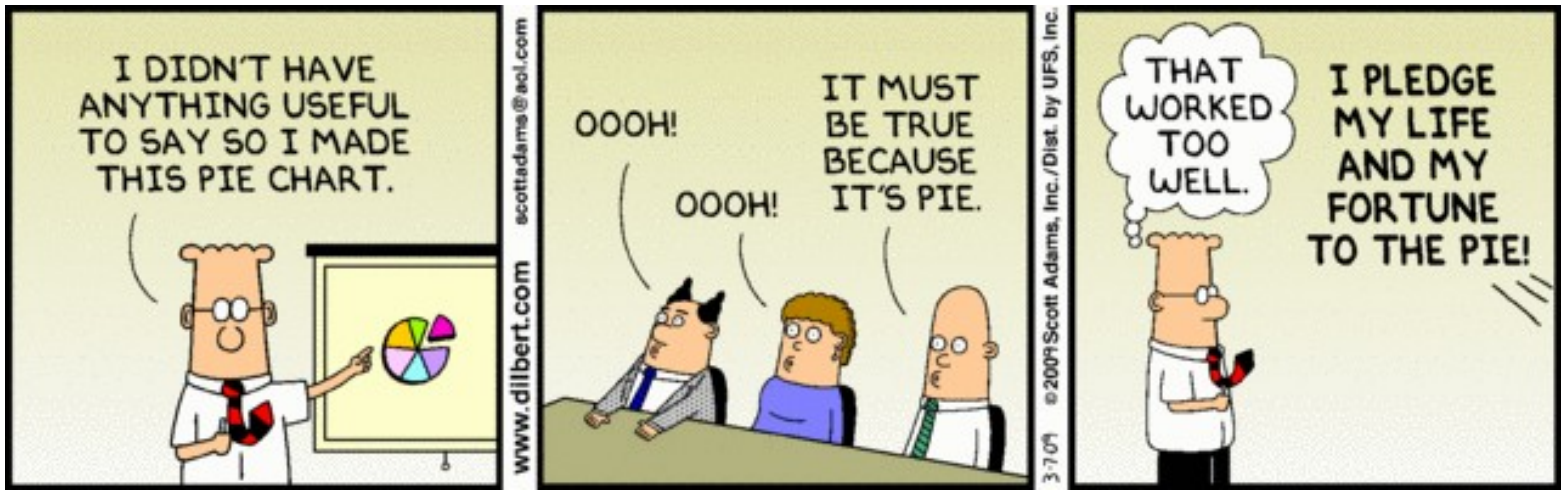
Sonar in numbers

5,000	downloads per month (from 2000 in 2009)
1000+	subscribers to mailing lists
50+	extensions in the forge
15	releases in 2 years
X?,000	running instances

Demo

Sonar is only a tool !

- What should happen in case new defects are added ?
- How, when and who should make quality standards evolve ?
- How to train new joiners ?
- Any measure reported must be analysed



The « Done, Done, Done, Done »

- Developed
- Tested
- Approved by the « Product Owner »
- **Technical debt under control**

Roadmap 2011

Bridge Internal / External Quality

Expand rules and metrics

Sonar IDE

Code Review

Track changes

C#

Developer Activity

Sonar-cpd

Questions & Answers

Thank You !

http://www.sonarsource.org
http://www.sonarsource.com

sonar