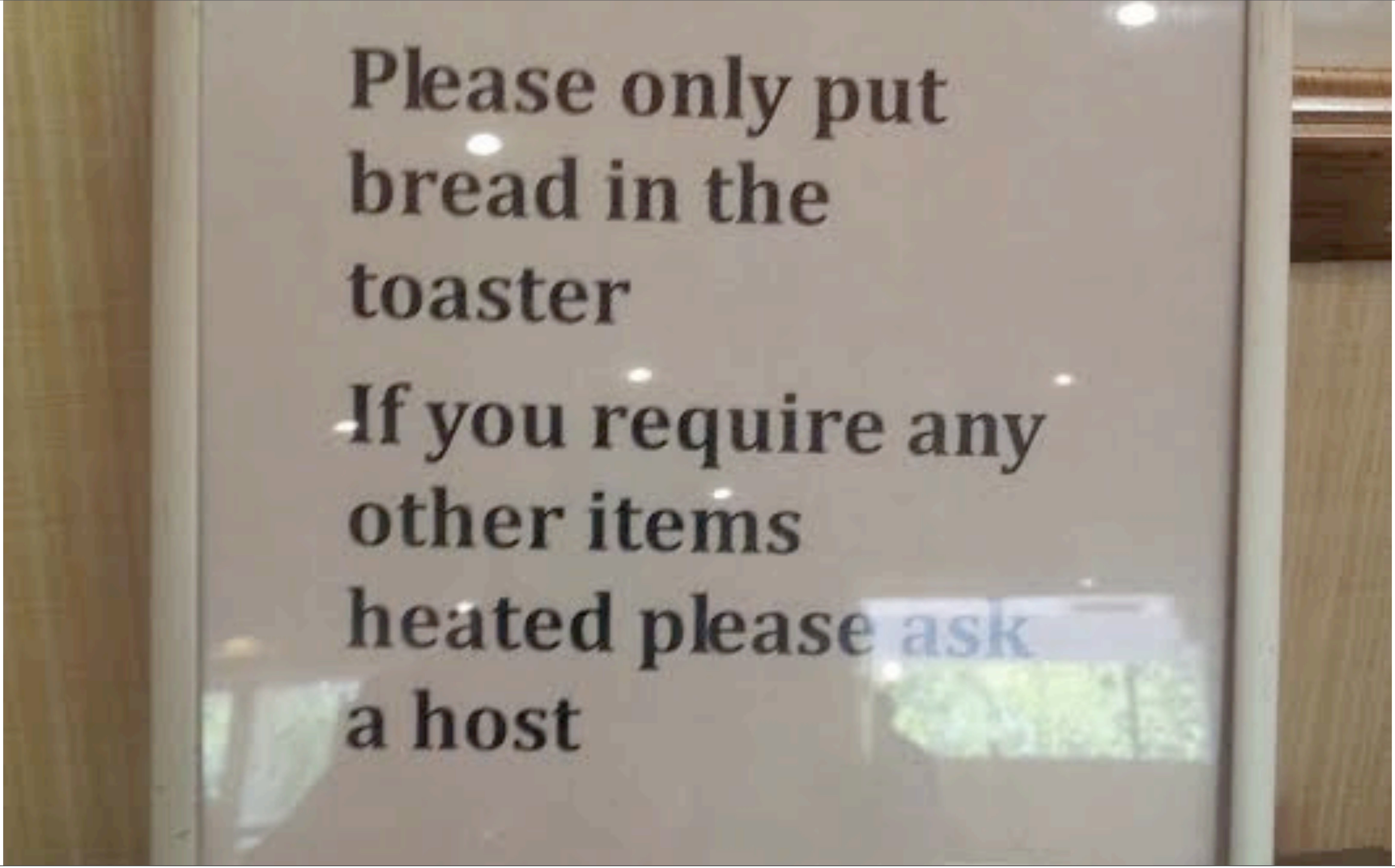
Making the Case for Review

Science & Practice

Austin Bingham

@austin_bingham



Please only put bread in the toaster

If you require any other items heated please ask a host

Introduction

What is review, and what is it good for?

1

What do we know about reviews?

A short review of some of the science about reviews

2

The benefits of review

How reviews can improve your software

3

The perils of review

When reviews attack!

4

Introducing reviews

How to start with reviews in your workflow

5

What is review?

“To view, look at, or look over again.”

- or -

“To inspect, especially formally or officially.”

dictionary.com

What is review?

Some act of looking over the work of yourself or another.

What is review?

Some act of looking over the work of yourself or another.

- Validation

What is review?

Some act of looking over the work of yourself or another.

- Validation
- Learning

What is review?

Some act of looking over the work of yourself or another.

- Validation
- Learning
- Quality check

What is review?

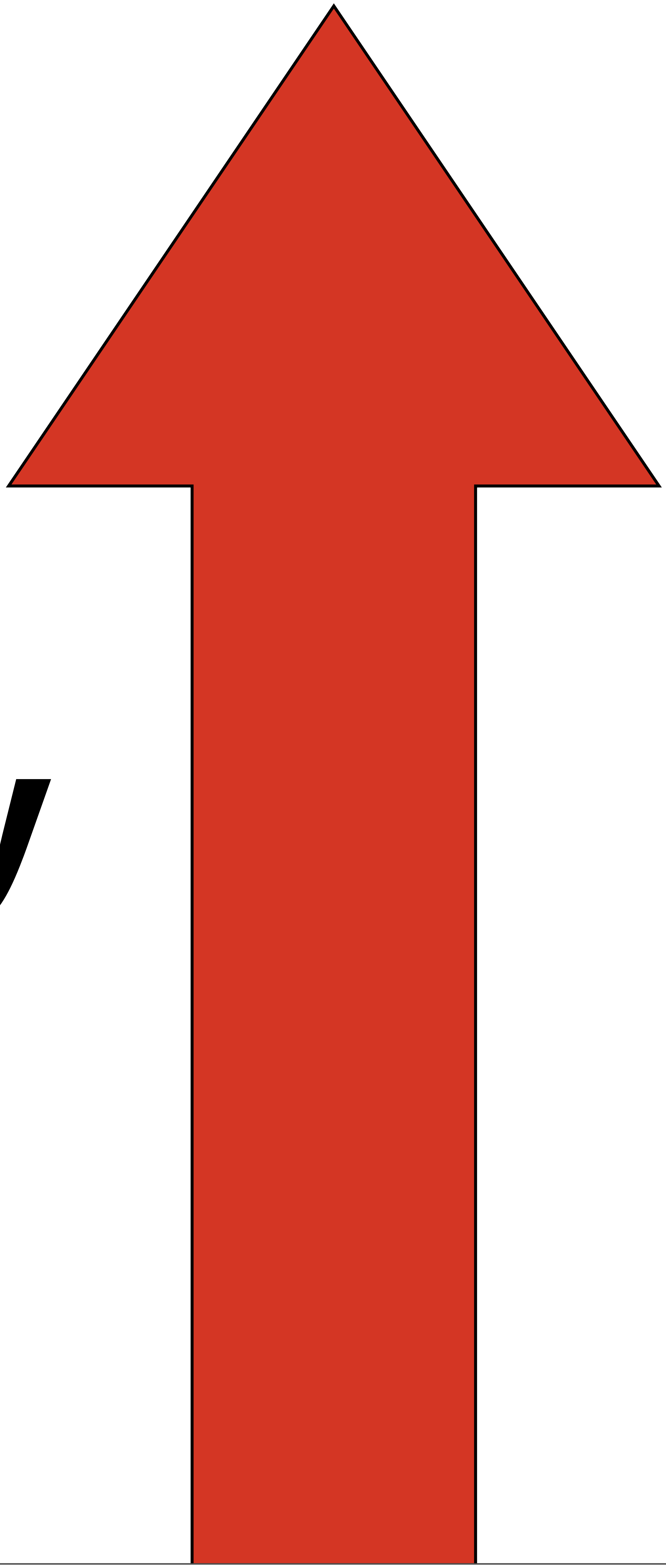
Some act of looking over the work of yourself or another.

- Validation
- Learning
- Quality check
- Whatever!

Improve **quality**

Improve **productivity**

Improve **teams**



Introduction

What is review, and what is it good for?

1

What do we know about reviews?

A short review of some of the science about reviews

2

The benefits of review

How reviews can improve your software

3

The perils of review

When reviews attack!

4

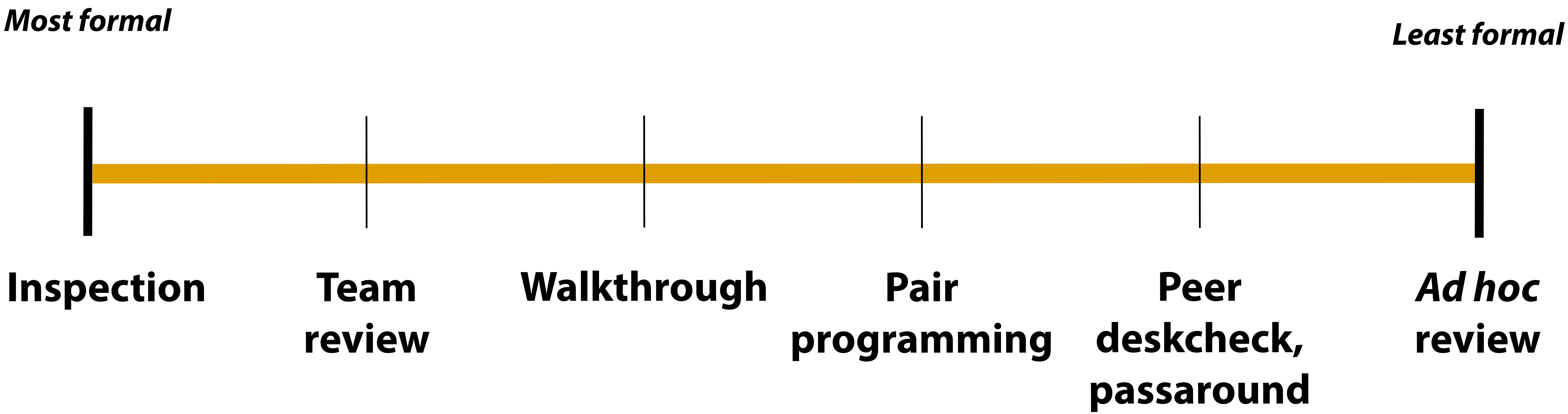
Introducing reviews

How to start with reviews in your workflow

5

Review formality spectrum

Reviews can be roughly ordered from formal inspections to *ad hoc*



Most formal

Least formal

Inspection

Team review

Walkthrough

Pair programming

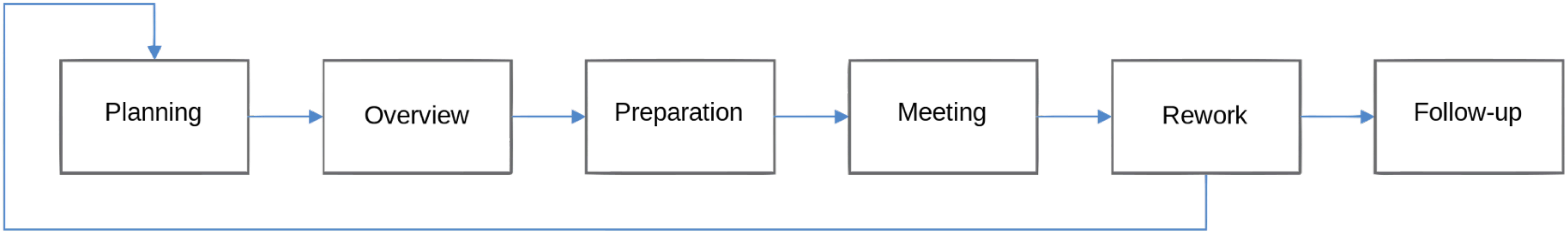
Peer deskcheck, passaround

***Ad hoc* review**

Based on the original diagram by Karl E. Wiegers in "Peer Reviews in Software: A Practical Guide"

Michael Fagan, 1976, IBM

Formal reviews / inspections



Meetings

Roles

Process

Data collection

Metrics

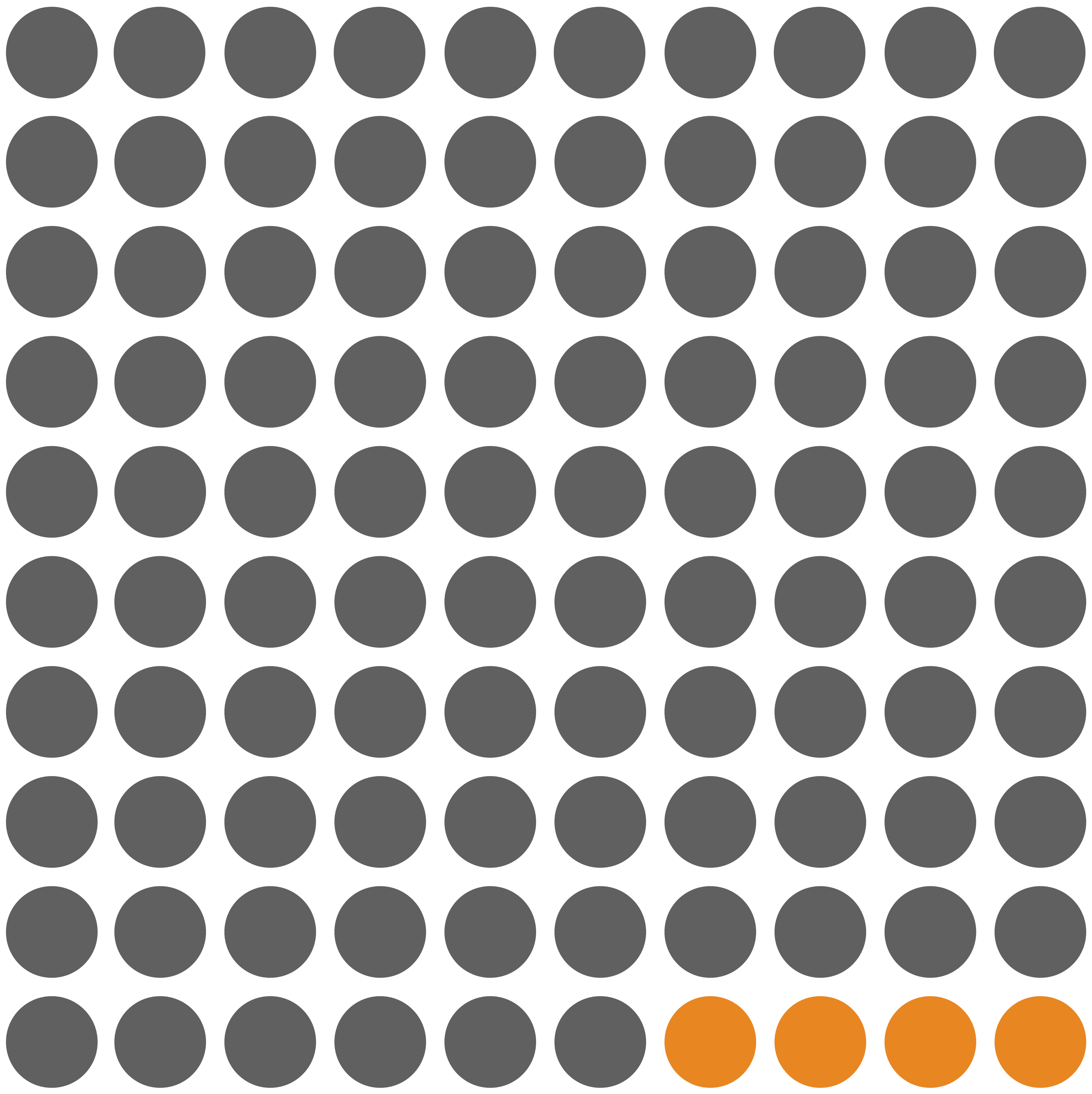
Lawrence Votta, 1993, Bell Labs

Are meetings really necessary for design reviews?

	Meetings		No Meetings
Synergy	Teams find faults better than individuals	→	Meetings tend to find false-positives
Education	Less-experienced learn from more-experienced	→	“Education by observation” is not very effective
Deadline	Meetings impose a schedule	→	Deadlines can be imposed without meetings
Competition	Egos give incentives to contribute/improve	→	Competition can be achieved without meetings
Process	“Inspections are part of official process.”	→	Facts, not tradition, should determine process

Lawrence Votta, 1993, Bell Labs

Are meetings really necessary for design reviews?



4%

of defects found in meetings

Diane Kelly & Terry Shepard, 2003, RMCC

Compare groups vs. individual for code reviews

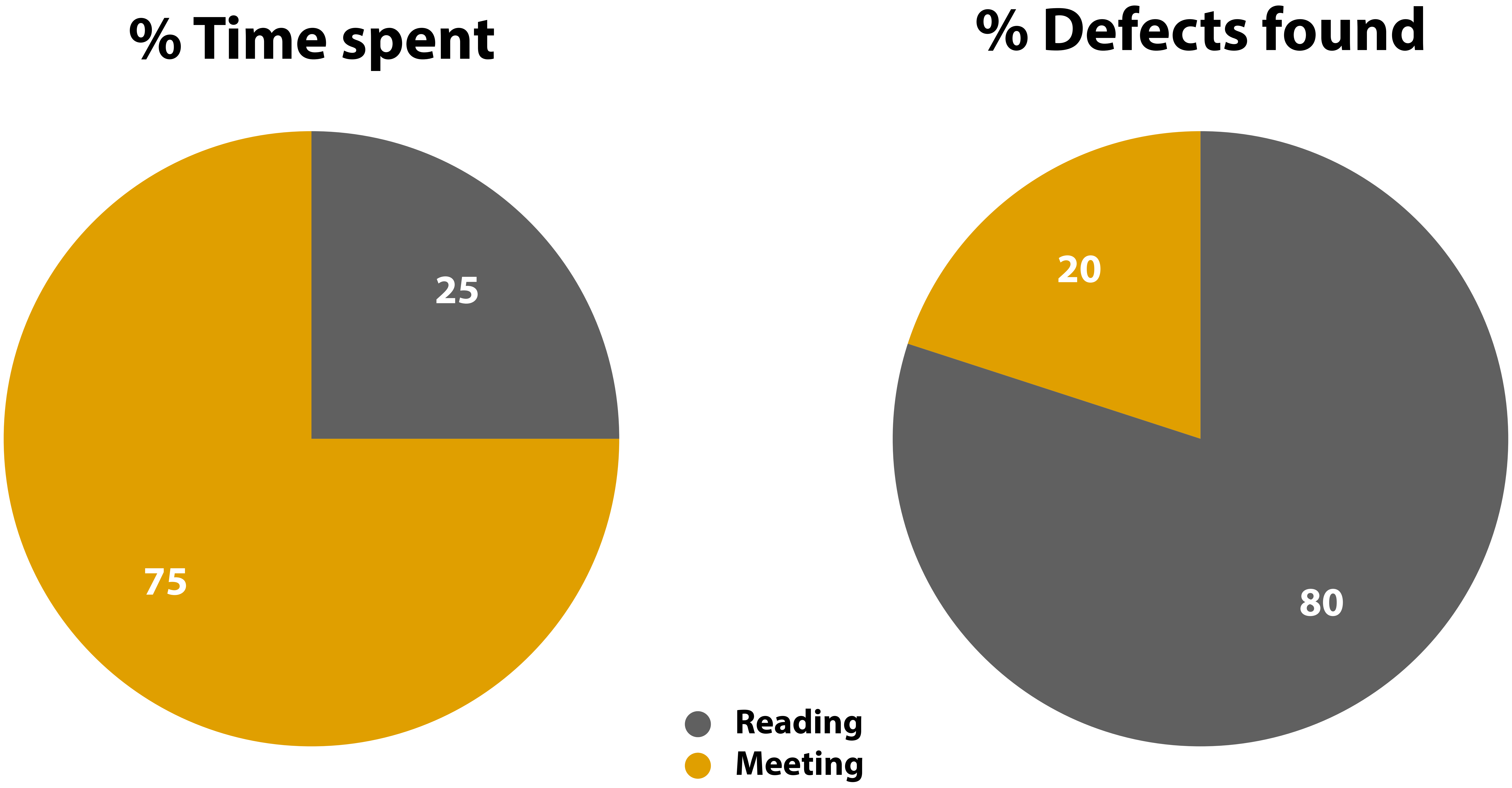
Largely confirmed Votta's findings.

Reading	Meeting
1.7 defects/hr.	1.2 defects/hr.

Reading is 50% more efficient

Reidar Conradi, 2003, Ericsson Norway/NTNU/Agder Univ.

Measure impact of reading techniques on UML inspections



Smartbear, 2006, Cisco

Large study of use of lightweight, tool-driven code review

Smartbear, 2006, Cisco

Large study of use of lightweight, tool-driven code review

- **Review size should be under 200, and no more than 400**

Smartbear, 2006, Cisco

Large study of use of lightweight, tool-driven code review

- **Review size should be under 200, and no more than 400**
- **Less than 300 LOC/hour for best detection rate**

Smartbear, 2006, Cisco

Large study of use of lightweight, tool-driven code review

- **Review size should be under 200, and no more than 400**
- **Less than 300 LOC/hour for best detection rate**
- **Author preparation/annotation results in far fewer defects**

Smartbear, 2006, Cisco

Large study of use of lightweight, tool-driven code review

- **Review size should be under 200, and no more than 400**
- **Less than 300 LOC/hour for best detection rate**
- **Author preparation/annotation results in far fewer defects**
- **Total review time should be less than 60 min., not to exceed 90 min.**

Smartbear, 2006, Cisco

Large study of use of lightweight, tool-driven code review

- **Review size should be under 200, and no more than 400**
- **Less than 300 LOC/hour for best detection rate**
- **Author preparation/annotation results in far fewer defects**
- **Total review time should be less than 60 min., not to exceed 90 min.**
- **Expect around 15 defects per hour**

Smartbear, 2006, Cisco

Large study of use of lightweight, tool-driven code review

- **Review size should be under 200, and no more than 400**
- **Less than 300 LOC/hour for best detection rate**
- **Author preparation/annotation results in far fewer defects**
- **Total review time should be less than 60 min., not to exceed 90 min.**
- **Expect around 15 defects per hour**
- **Inspection rates can vary widely**

Smartbear, 2006, Cisco

Large study of use of lightweight, tool-driven code review

- **Review size should be under 200, and no more than 400**
- **Less than 300 LOC/hour for best detection rate**
- **Author preparation/annotation results in far fewer defects**
- **Total review time should be less than 60 min., not to exceed 90 min.**
- **Expect around 15 defects per hour**
- **Inspection rates can vary widely**

Review between 100 and 300 LOC

Spend 30-60 minutes

Spend at least 5 minutes for even a single-line review

Meeting are good for finding

false-positives

so keep them

short and small

Cost saving from reviews

As reported in “Peer Reviews in Software”, Wiegers

Hewlett-Packard	10:1 ROI, saving $21.4 million per year.
AT&T Bell Labs	Error-detection cost reduced by a factor of 10. 10-fold quality improvement. 14% productivity increase.
Bell Northern Research	Prevented 33 hours of maintenance per defect discovered. 2-4x speed detection-time improvement versus testing.
IBM	1 hour of inspection saved 20 hours of testing and 82 hours of rework (if defect had made it to customers.)
Imperial Chemical	Maintenance cost for inspected programs was 1/10th of that for uninspected programs.
Litton Data Systems	3% project effort in inspections reduced testing defects by 30%. Design and code inspections cut integration effort in half.

Upstream inspection is powerful

Finding defects in early phases avoids wasted work in later phases

"Bellcore found that **44 percent** of all bugs were due to **defects in requirements and design** reaching the programmers."

Tom Gilb, *Optimizing Software Inspections*

Upstream inspection is powerful

Finding defects in early phases avoids wasted work in later phases

"Bellcore found that **44 percent** of all bugs were due to **defects in requirements and design** reaching the programmers."

Tom Gilb, *Optimizing Software Inspections*

Programmers can do whatever you ask them to do!

"Research study after research study has shown that inspections can detect up to 90% of the errors in a software product before any test cases have been run. And that signifies an extremely effective process."

Robert Glass

"...the same studies show that the cost of inspections is less than the cost of the testing that would be necessary to find the same errors. What we have here is an effective process that is also cost-effective. And that's a pretty nice combination."

Robert Glass

What about testing?

Frank Blakely et al., 1991, HP

Cost-effectiveness of inspection vs. testing

21

defects found in inspection

Frank Blakely et al., 1991, HP

Cost-effectiveness of inspection vs. testing

21

defects found in inspection

4

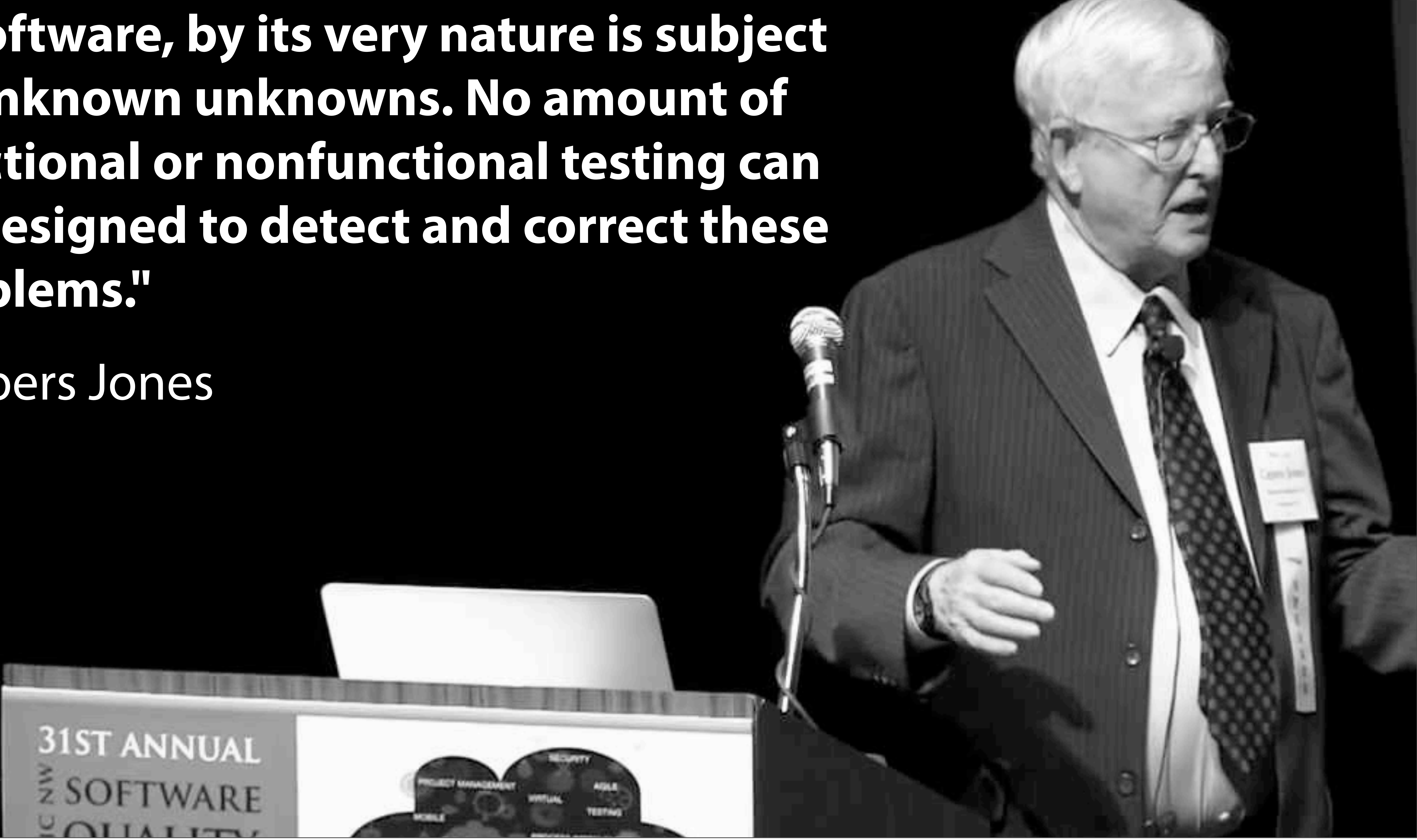
would have been found in testing

"Testing alone has never been sufficient to achieve high-quality software."

- Capers Jones

"...software, by its very nature is subject to unknown unknowns. No amount of functional or nonfunctional testing can be designed to detect and correct these problems."

- Capers Jones



Want to know more?

Best Kept Secrets of Peer Code Review

Modern Approach. Practical Advice.

Jason Cohen

Addison-Wesley Information Technology Series

Peer Reviews in Software

A Practical Guide

Karl E. Wiegers

Facts and Fallacies of Software Engineering

Robert L. Glass

Foreword by Alan M. Davis

The Economics of Software Quality

Capers Jones

vier Bonsignour

eword by Thaddeus Arroyo

nation Officer, AT&T Services, Inc.

Want to know more?

Want to know more?

Best Kept Secret
of
Peer Code Revi
Modern App
Practical A
Jason Cohen
Addison-Wesley Information Technology Series
Peer
Reviews
in Software
A Practical Guide
Karl E. Wiegers
The Economics
of Software Quality
nes
gnour
us Arroyo
&T Services, Inc.
s and Fallacies of
ware Engineering
Robert L. Glass
Foreword by Alan M. Davis
THE
LEPRECHAUNS
OF
SOFTWARE
ENGINEERING
HOW FOLKLORE TURNS INTO FACT AND WHAT TO DO ABOUT IT
LAURENT BOSSAVIT

Introduction

What is review, and what is it good for?

1

What do we know about reviews?

A short review of some of the science about reviews

2

The benefits of review

How reviews can improve your software

3

The perils of review

When reviews attack!

4

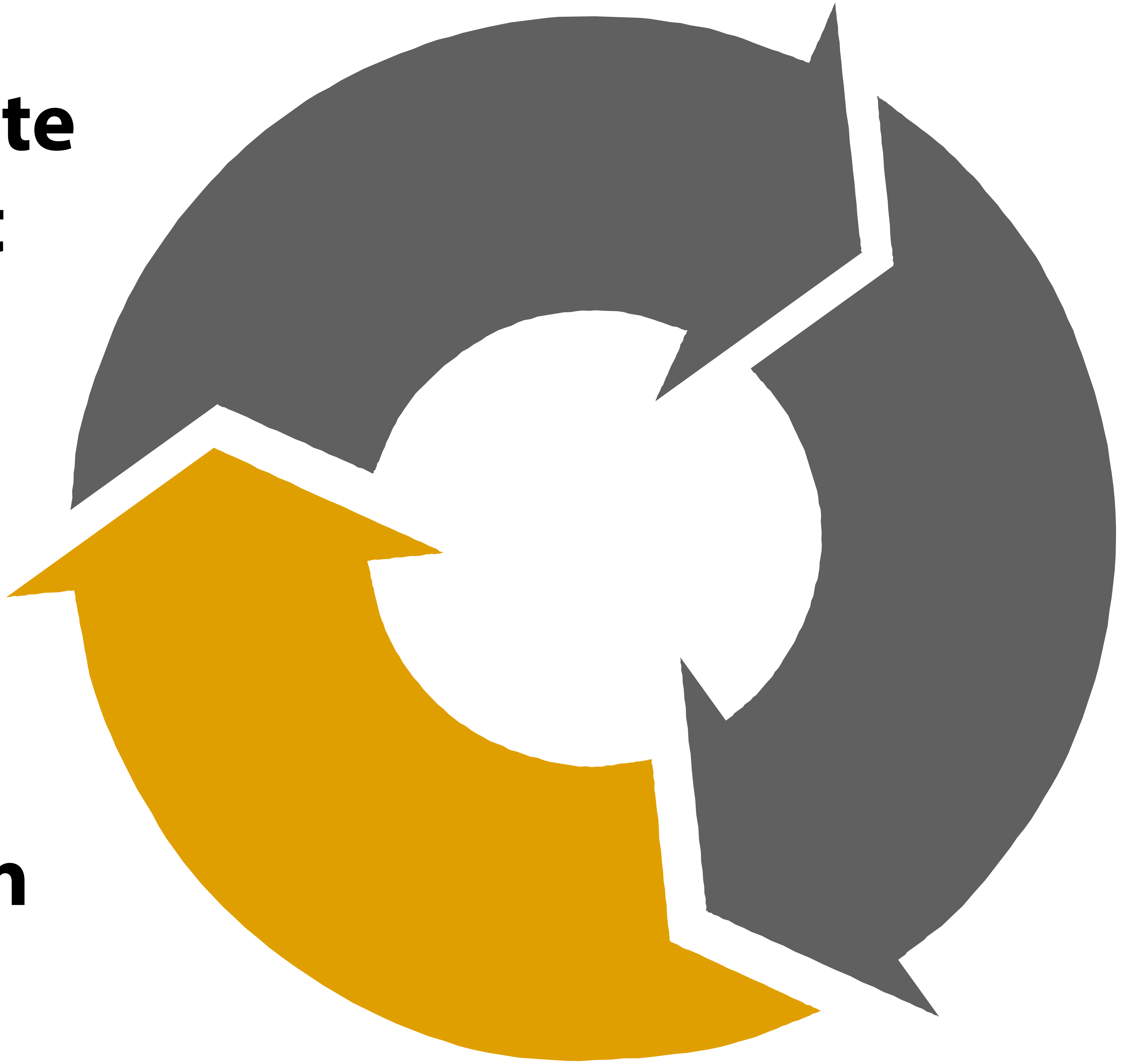
Introducing reviews

How to start with reviews in your workflow

5

Defect prevention

Reviews reduce defect injection rates



Generate artifact

Review artifact

Learn

Mentoring

Reviews provide plenty of "teachable moments"

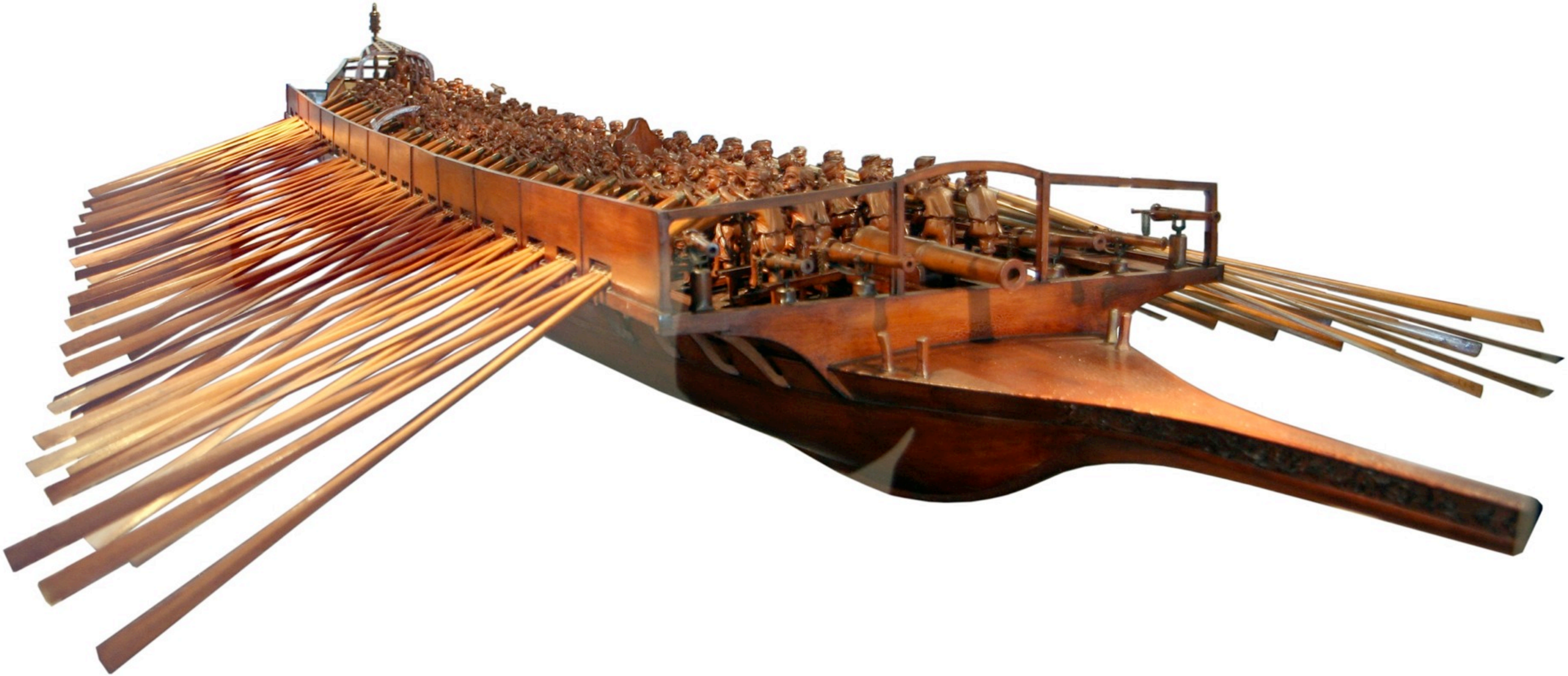
Monitoring and learning

Reviews allow you to see what others are doing

- **Code Quality**
- **Growth of junior members**
- **Habits of senior members**
- **New ideas and techniques**

Team cohesion

Shared experience and group ownership



Confidence

Part of the record

Review tools can be helpful for recording decisions

All Projects Documentation
Open Merged Abandoned
android
open source project

Change 83824 - **Merged**

Reply...

```
Make executeHardwareAction avaiable to AwContents

Removed wrapping all of onDetachedFromWindow with executeHardwareAction,
since upstream code will have finer control on what code to wrap
executeHardwareAction in.

Also take this opportunity to clean up native public interface.

This is the first of a 3-sided patch. The upstream change is at
https://codereview.chromium.org/185133003/

Change-Id: Ie791842a0becd8189dd8d52a027b75b8ee34211a
```

Owner	Bo Liu
Reviewers	Bart Sears Selim Gurun Bart Sears Ben Murdoch Selim Gurun
Project	platform/frameworks/webview
Branch	chromium-dev
Topic	
Updated	13 hours ago

Code-Review	+2	Selim Gurun
Verified	+1	Bart Sears

Author	Bo Liu <boliu@google.com>	Feb 28, 2014 4:14 PM
Committer	Bo Liu <boliu@google.com>	Mar 3, 2014 10:54 PM
Commit	55aff944126d8f2dc4020ec88405a93919579db2	(gitiles)
Change-Id	Ie791842a0becd8189dd8d52a027b75b8ee34211a	

Files Open All Diff against: Base

File Path	Comments	Size
▸ Commit Message		
chromium/java/com/android/webview/chromium/WebViewChromium.java		17
chromium/plat_support/draw_gl_functor.cpp		23
chromium/plat_support/graphics_utils.cpp		26
		+7, -59

History Expand All

Bo Liu	Uploaded patch set 1.	Mar 2 6:17 AM
Bo Liu	Uploaded patch set 2.	Mar 3 6:42 PM
Bo Liu	Uploaded patch set 3.	Mar 3 10:09 PM
Bo Liu	Uploaded patch set 4.	Mar 3 10:55 PM
Selim Gurun	Patch Set 4: Code-Review+2	Mar 4 1:51 AM
Bo Liu	Patch Set 4: Chromium side is ready to go, so want to subimt this. bsears@ could you give +1 verify?	Mar 4 9:15 PM
Bart Sears	Patch Set 4: Verified+1	Mar 4 10:10 PM
Bart Sears	Change has been successfully merged into the git repository.	12:30 AM

Defect reduction

Peer reviews are an excellent way to find defects early in your process

"Peer review catches

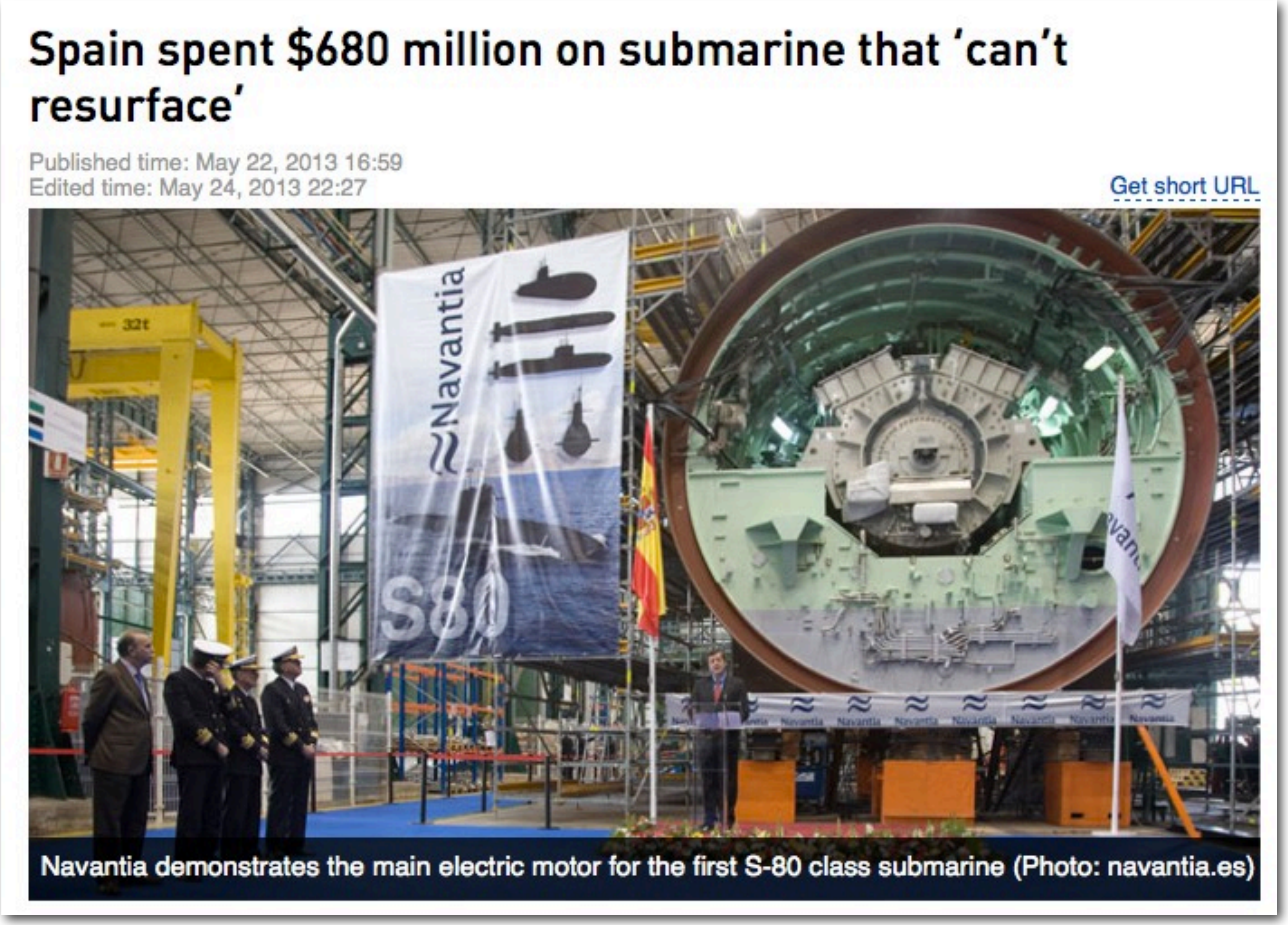
60%

of the defects."

Boehm, Basili, http://www.cs.umd.edu/projects/SoftEng/ESEG/papers/82.78.pdf

Diminish effects of ego

Everybody screws up sometimes!



Spain spent $680 million on submarine that ‘can’t resurface’

Published time: May 22, 2013 16:59
Edited time: May 24, 2013 22:27

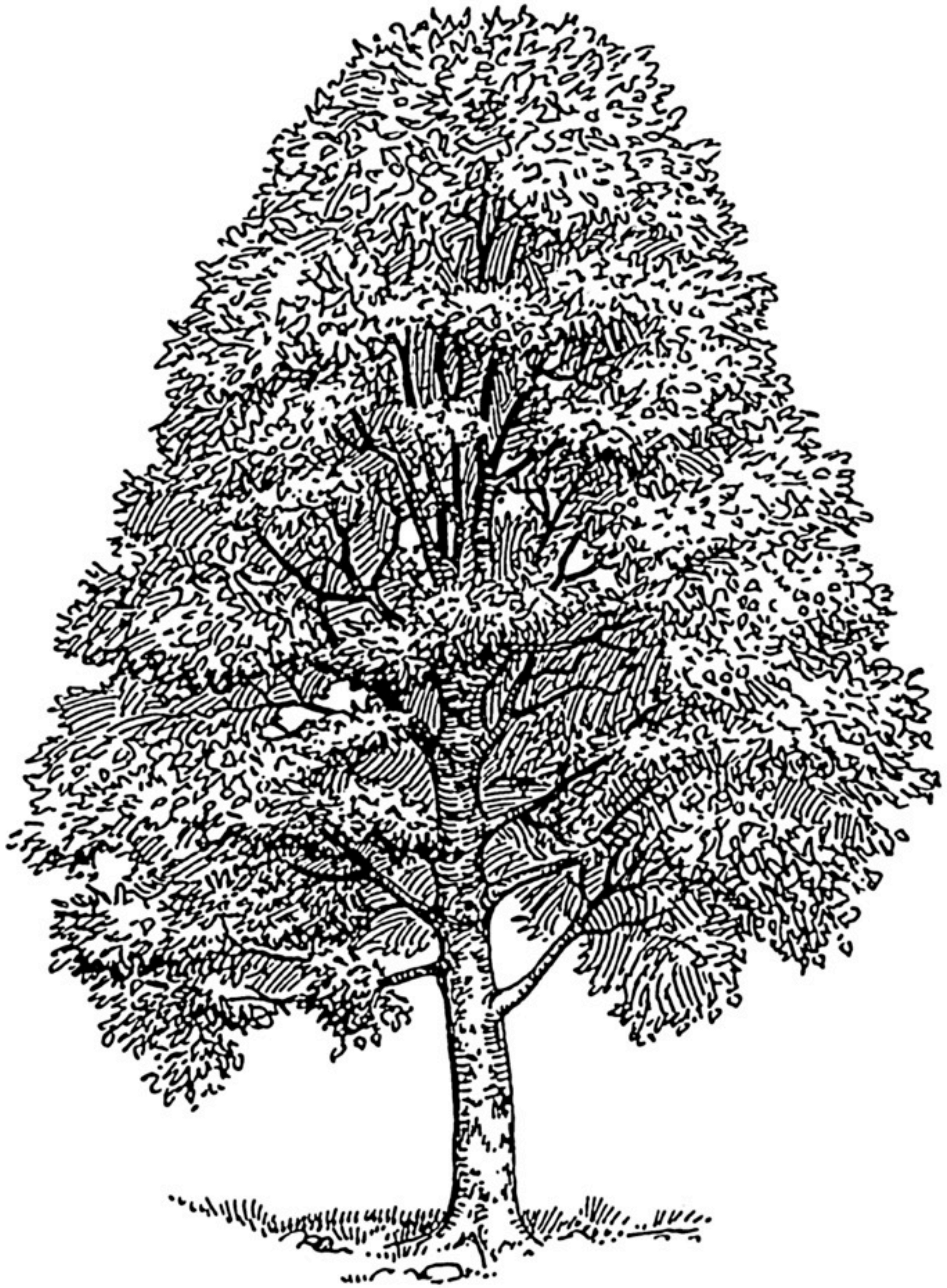
Get short URL

Navantia demonstrates the main electric motor for the first S-80 class submarine (Photo: navantia.es)

http://rt.com/news/spanish-submarine-cannot-resurface-634/

Personal growth

Review results can reveal patterns and bad practices that you can then fix.



Introduction

What is review, and what is it good for?

1

What do we know about reviews?

A short review of some of the science about reviews

2

The benefits of review

How reviews can improve your software

3

The perils of review

When reviews attack!

4

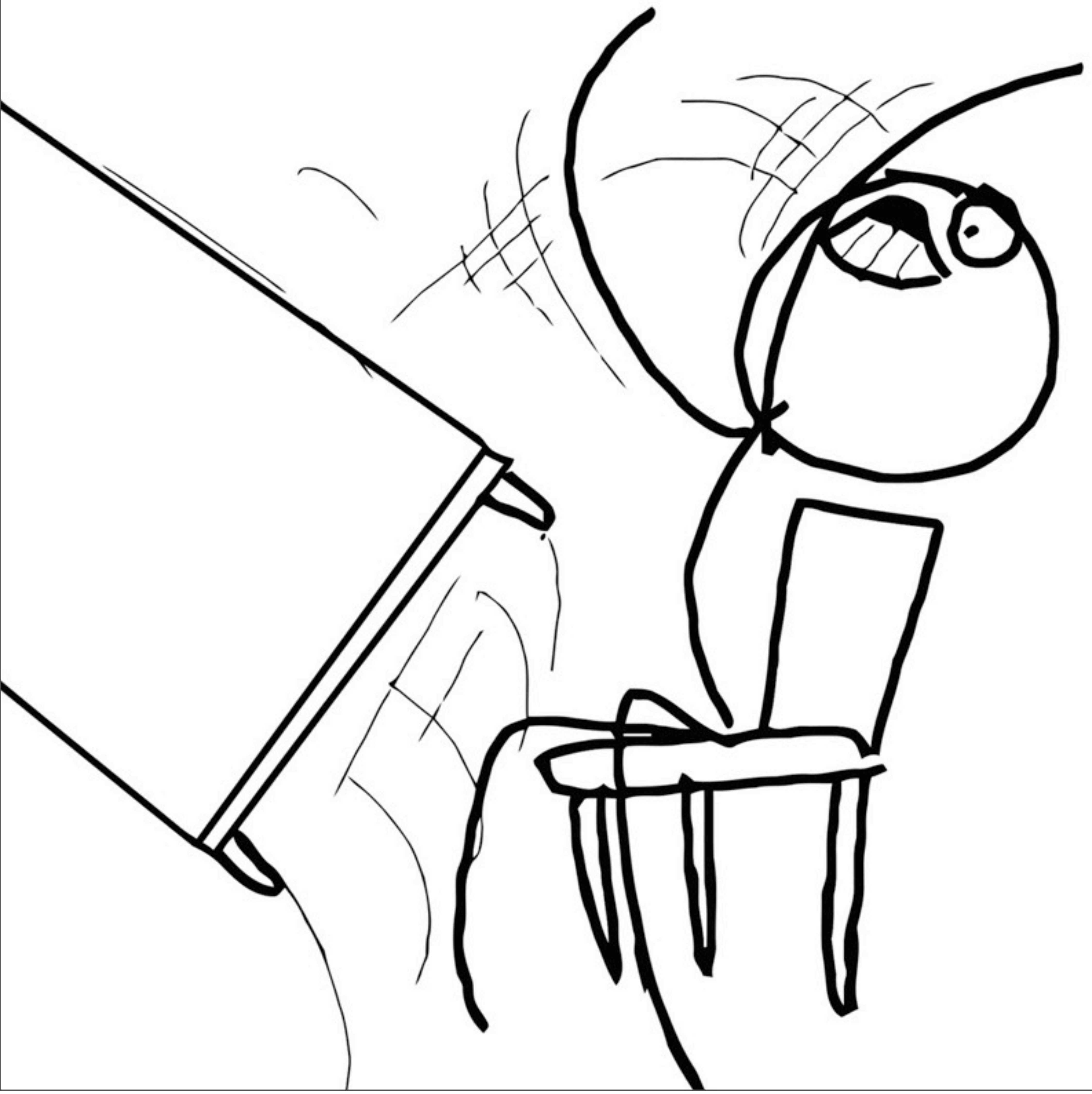
Introducing reviews

How to start with reviews in your workflow

5

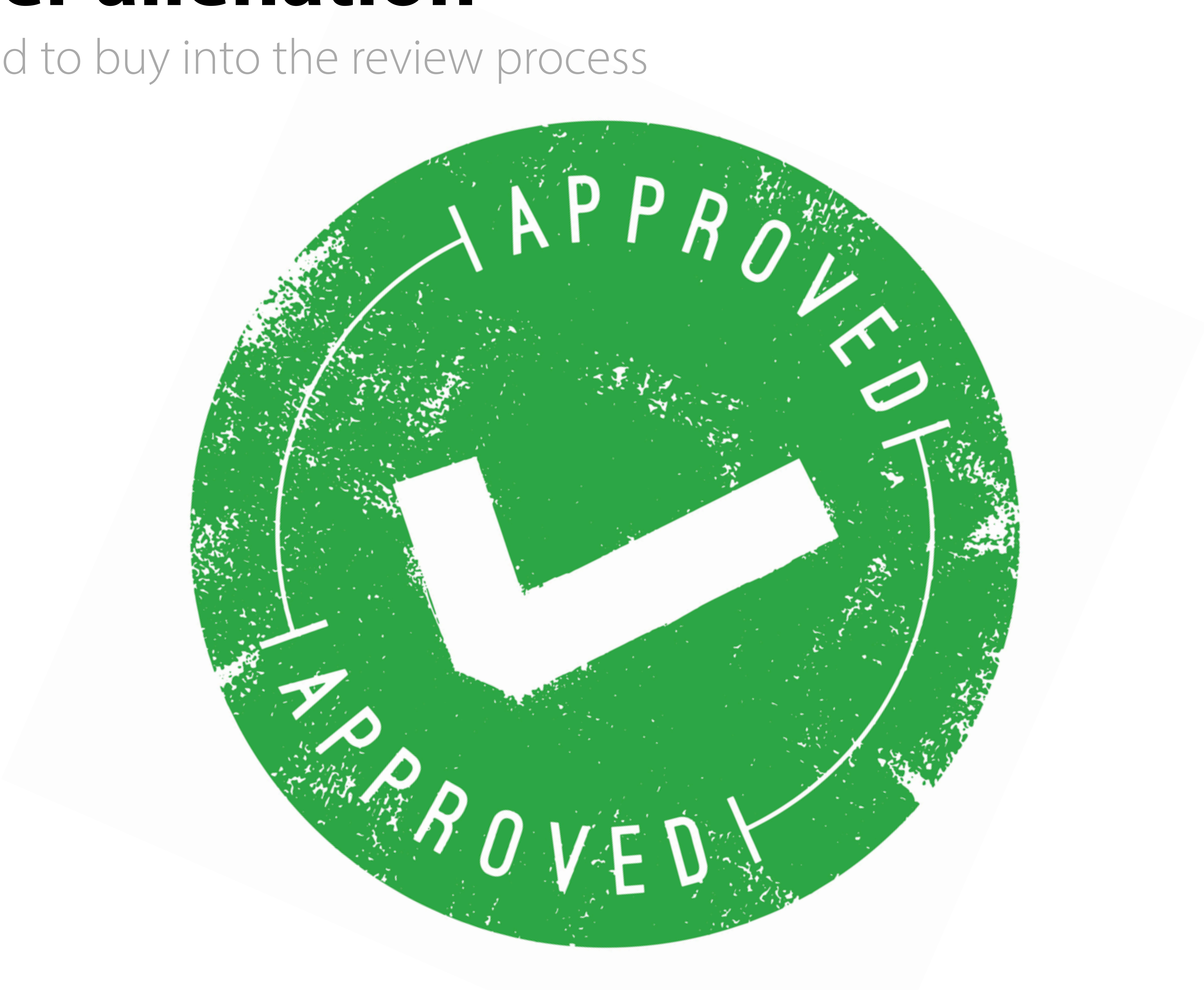
Egos

Reviews can also inflame egos if they're perceived as attacks



Developer alienation

Developers need to buy into the review process

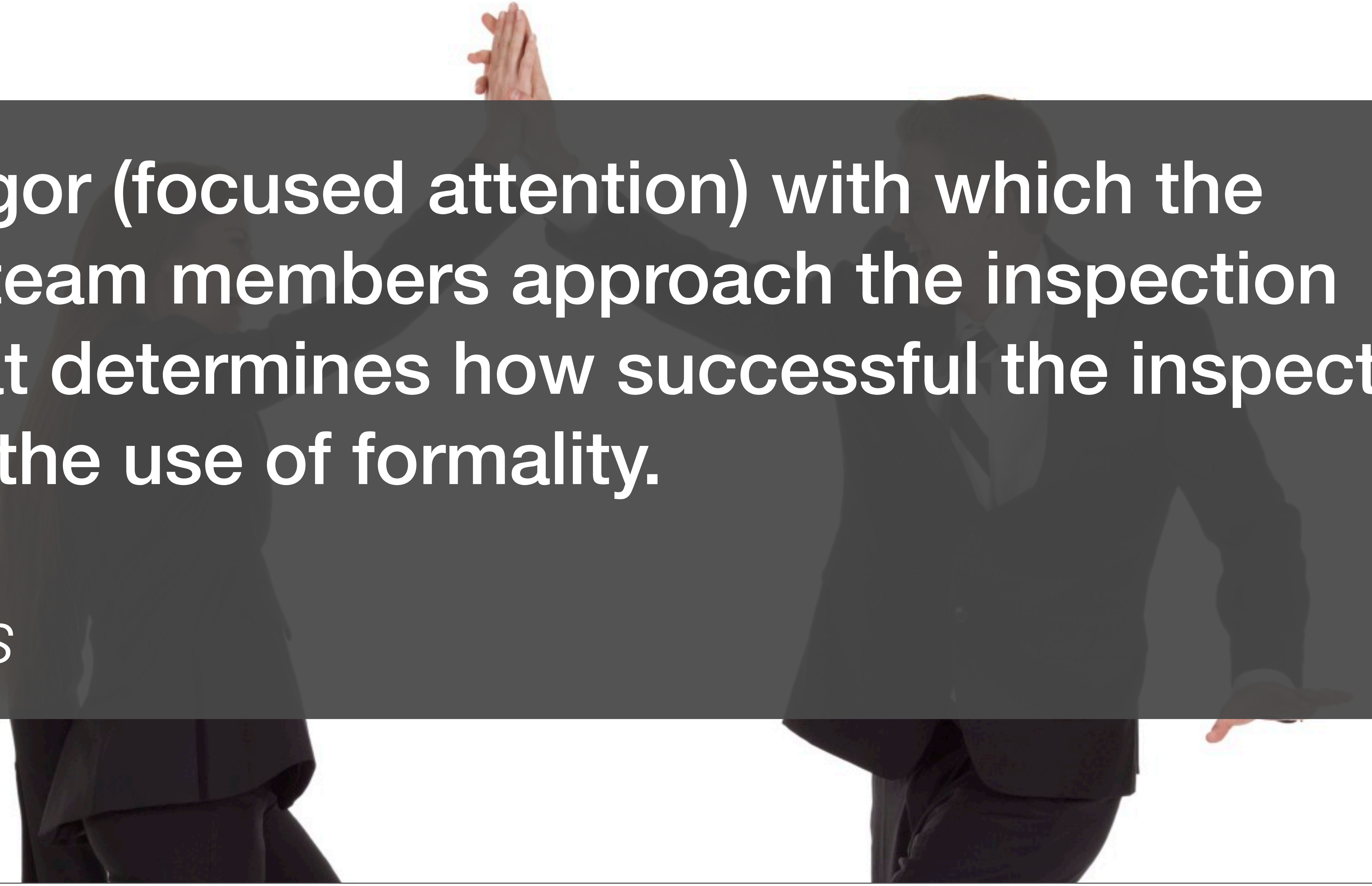


Wasted time

Uncritical or shallow reviews cost time and don't improve quality

Wasted time

Uncritical or shallow reviews cost time and don't improve quality

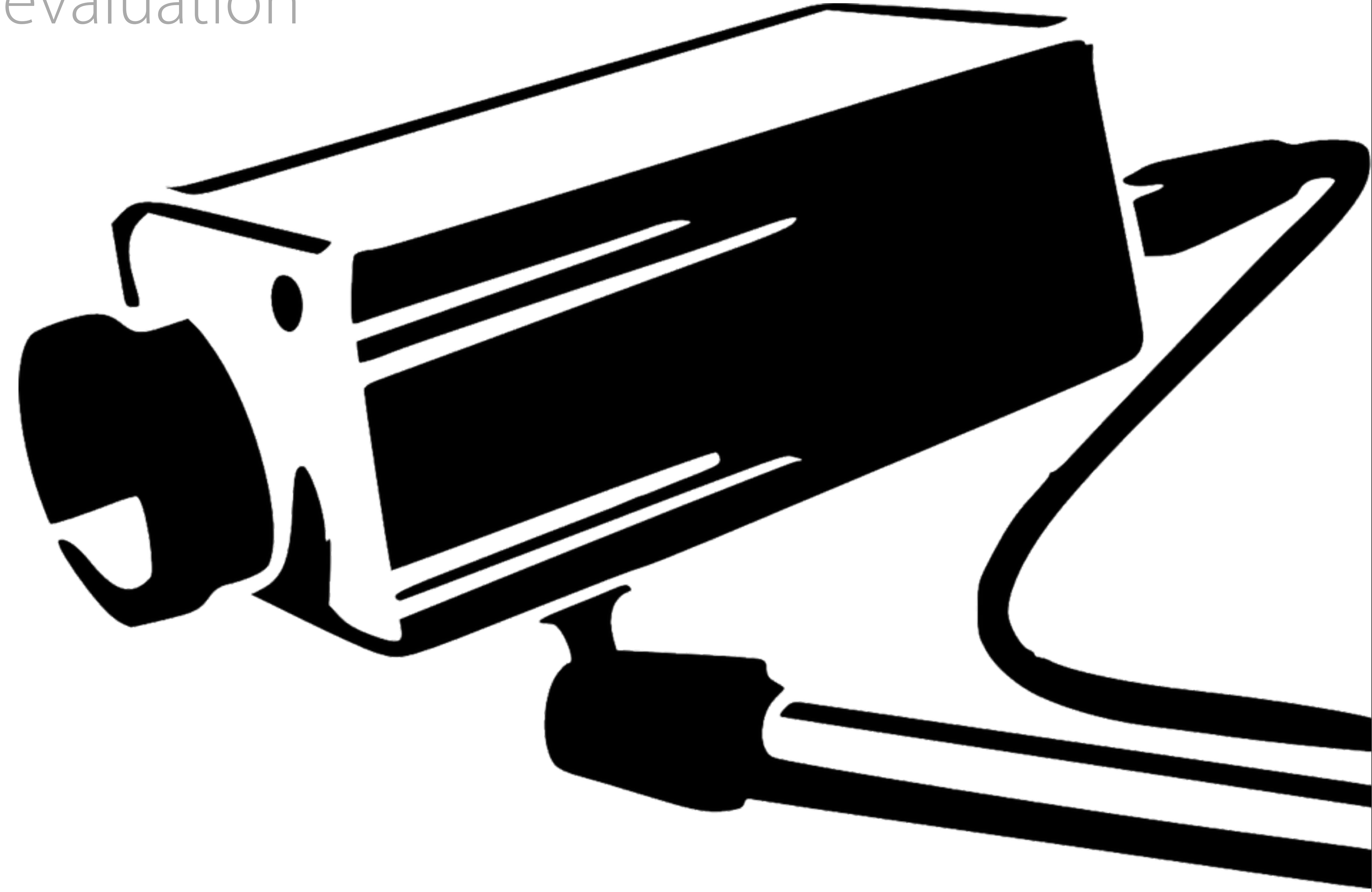


...it is the rigor (focused attention) with which the inspection team members approach the inspection process that determines how successful the inspection will be, not the use of formality.

Robert Glass

Big Brother effect

It is dangerous to tie review data to employee evaluation



"Tell me how you will measure me, and I will tell you how I behave."

- Eli Goldratt, "The Goal"

Flow disruption

Reviews can become a distraction

Introduction

What is review, and what is it good for?

1

What do we know about reviews?

A short review of some of the science about reviews

2

The benefits of review

How reviews can improve your software

3

The perils of review

When reviews attack!

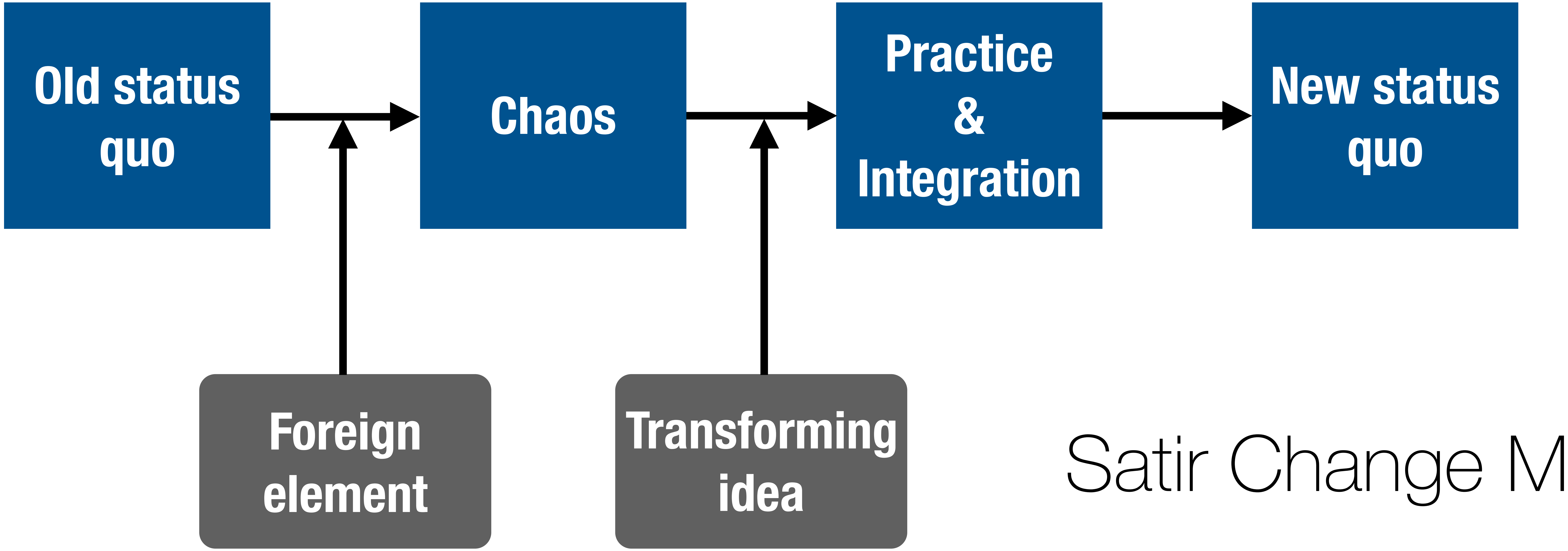
4

Introducing reviews

How to start with reviews in your workflow

5

Management support


Old status quo

Chaos

Practice & Integration

New status quo

Foreign element

Transforming idea

Satir Change Model

Selling reviews to management

Speak their language

Developer support

Make results tangible

Jenkins

Jenkins ▸ All ▸ openocd-gerrit-build ▸

- Back to Dashboard
- Status
- Changes
- Email Template Testing

Project openocd-gerrit-build

Configurations

🔵 linux32 🔵 mingw64 🔵 mingw32 🔵 armlinux 🔵 zy1000 🔵 distcheck

Build History			(trend)
🔵	#3547	Mar 5, 2014 9:04:08 PM	2019,5
🔵	#3546	Mar 5, 2014 5:45:16 AM	2018,2
🔵	#3545	Mar 4, 2014 10:55:41 PM	2019,4
🔴	#3544	Mar 4, 2014 10:06:37 PM	2019,3
🔵	#3543	Mar 4, 2014 9:52:55 PM	2020,1 🔒
🔴	#3542	Mar 4, 2014 9:42:19 PM	2019,2
🔴	#3541	Mar 4, 2014 9:22:54 PM	2019,1
🔵	#3540	Mar 4, 2014 5:56:19 PM	2018,1 🔒
🔵	#3539	Mar 4, 2014 4:58:59 PM	2000,2
🔵	#3538	Mar 4, 2014 9:14:31 AM	1427,7
🔵	#3537	Mar 3, 2014 9:34:26 PM	2017,1
🔵	#3536	Mar 3, 2014 9:21:40 PM	2016,1
🔵	#3535	Mar 3, 2014 9:07:50 PM	2015,1

Don't be too disruptive

“People hate change...
and that’s because people hate change...
I want to be sure that you get my point.
People really hate change.
They really, really do.”

Steve McMenamin, The Atlantic Systems Guild, 1996

But be disruptive enough!

Get everyone involved quickly

In Practice: Where to start?

In Practice: Where to start?

- Code reviews are the most obvious

In Practice: Where to start?

- Code reviews are the most obvious
- But start with what makes sense for you!

In Practice: Where to start?

- Code reviews are the most obvious
- But start with what makes sense for you!
- Increase coverage organically

In Practice: Maintenance

In Practice: Maintenance

- **Vigilance!**

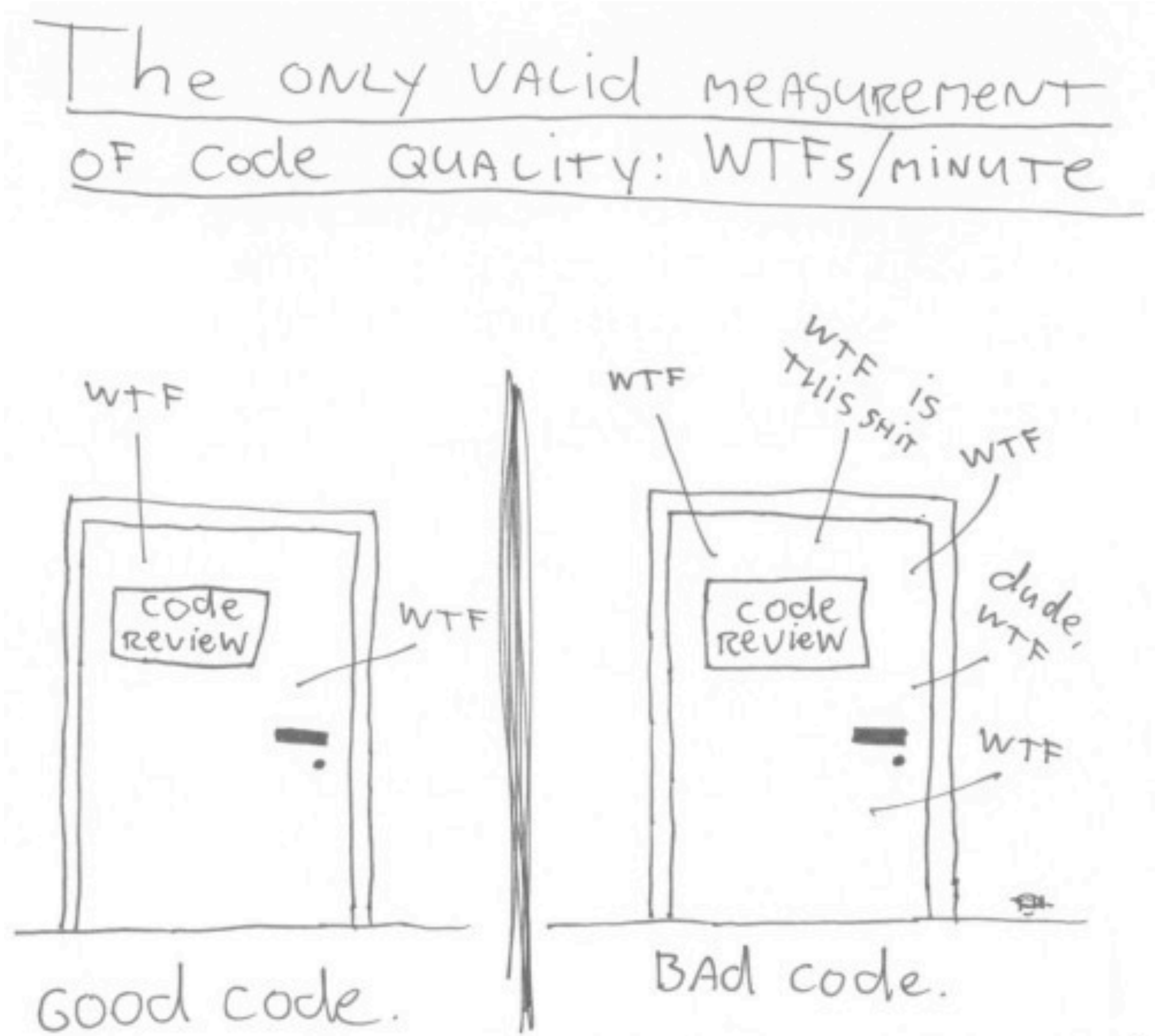
In Practice: Maintenance

- **Vigilance!**
- **Emphasize the benefits**

In Practice: Maintenance

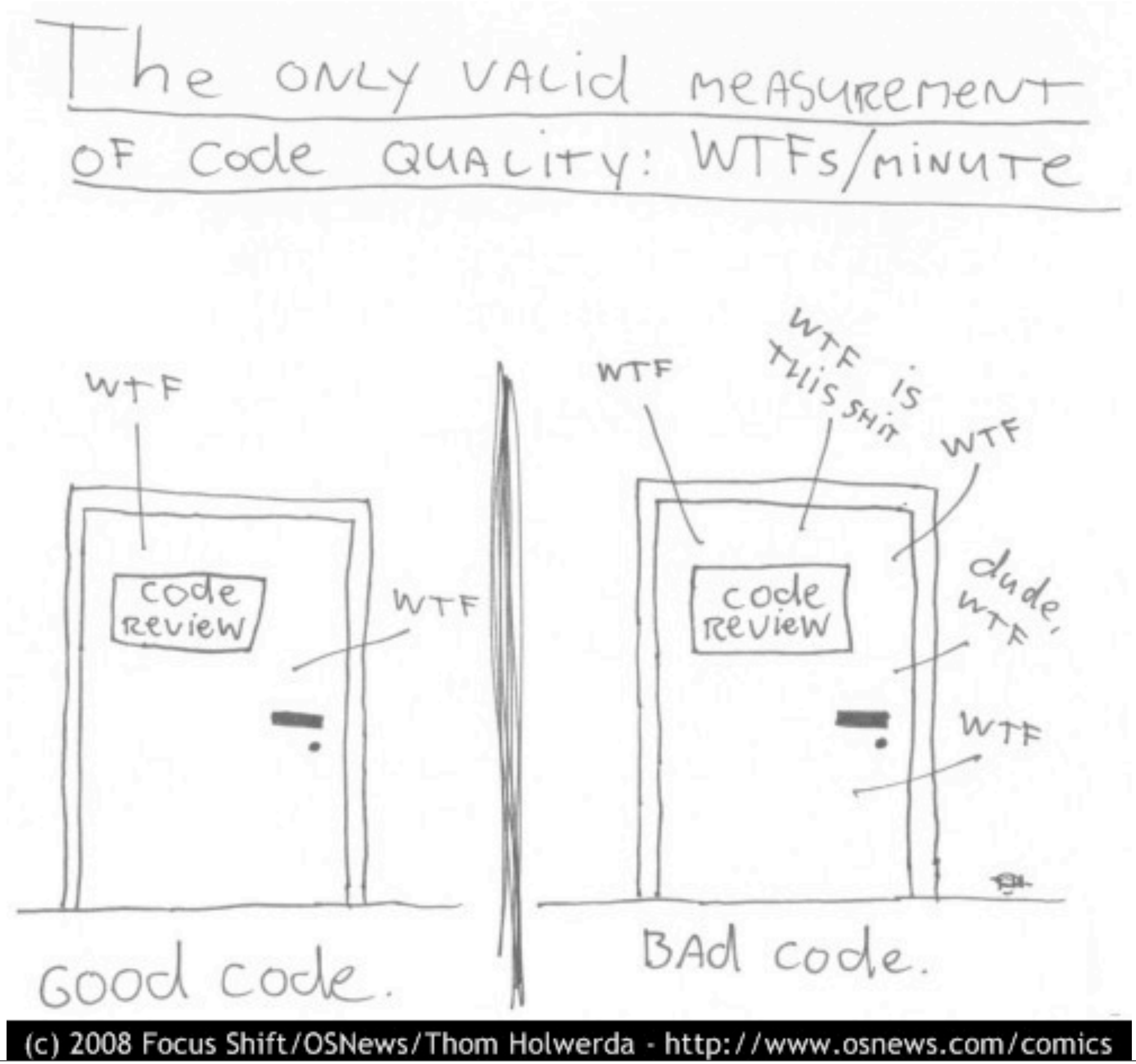
- Vigilance!
- Emphasize the benefits
- Avoid excessive ceremony

In Practice: What's in a review?



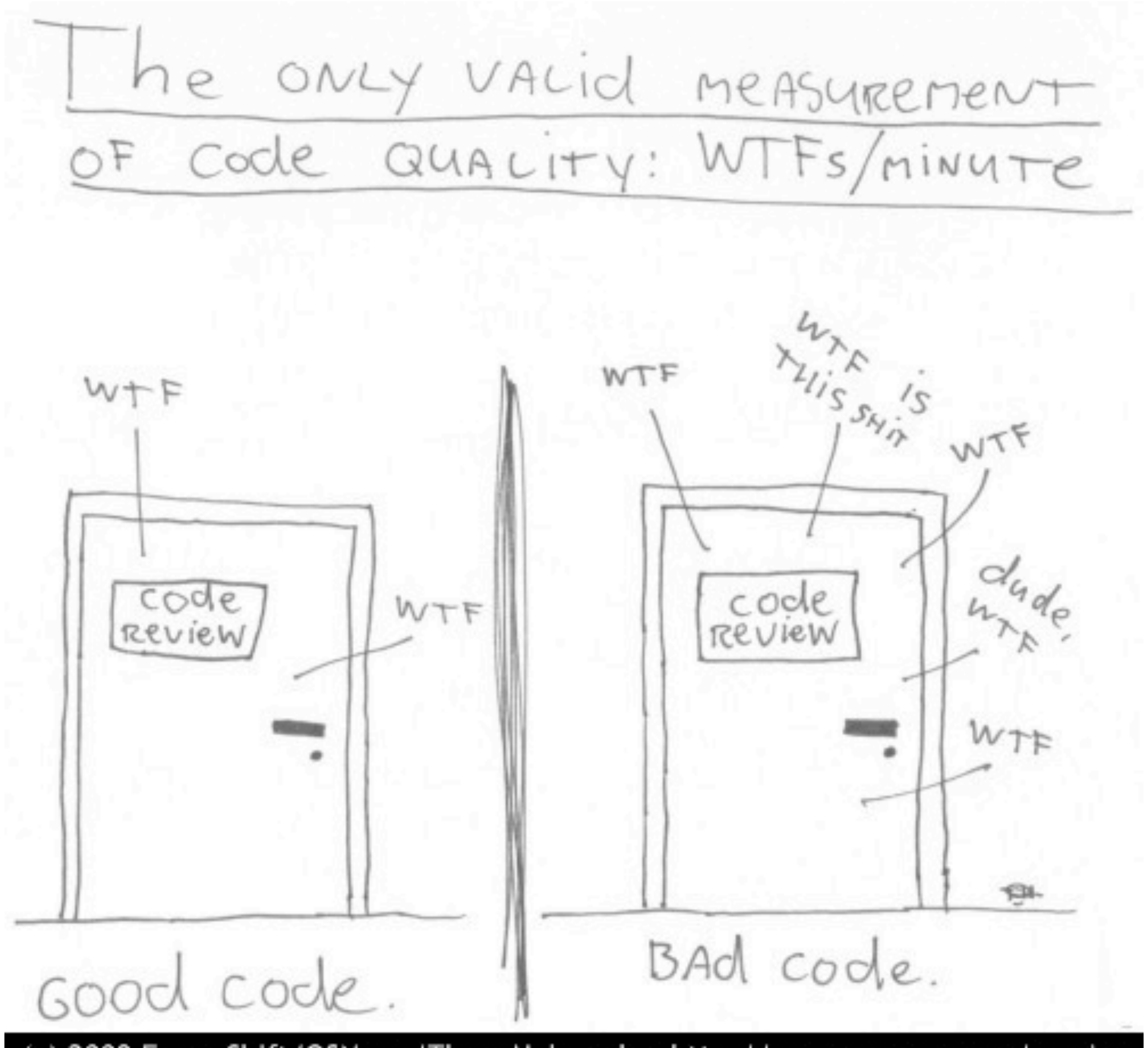
(c) 2008 Focus Shift/OSNews/Thom Holwerda - http://www.osnews.com/comics

In Practice: What's in a review?


The only valid measurement of code quality: WTFs/minute
WTF
code review
WTF
Good code.
WTF
WTF is this shit
WTF
code review
dude, WTF
WTF
BAd code.
(c) 2008 Focus Shift/OSNews/Thom Holwerda - http://www.osnews.com/comics

- **At least one competent reviewer**

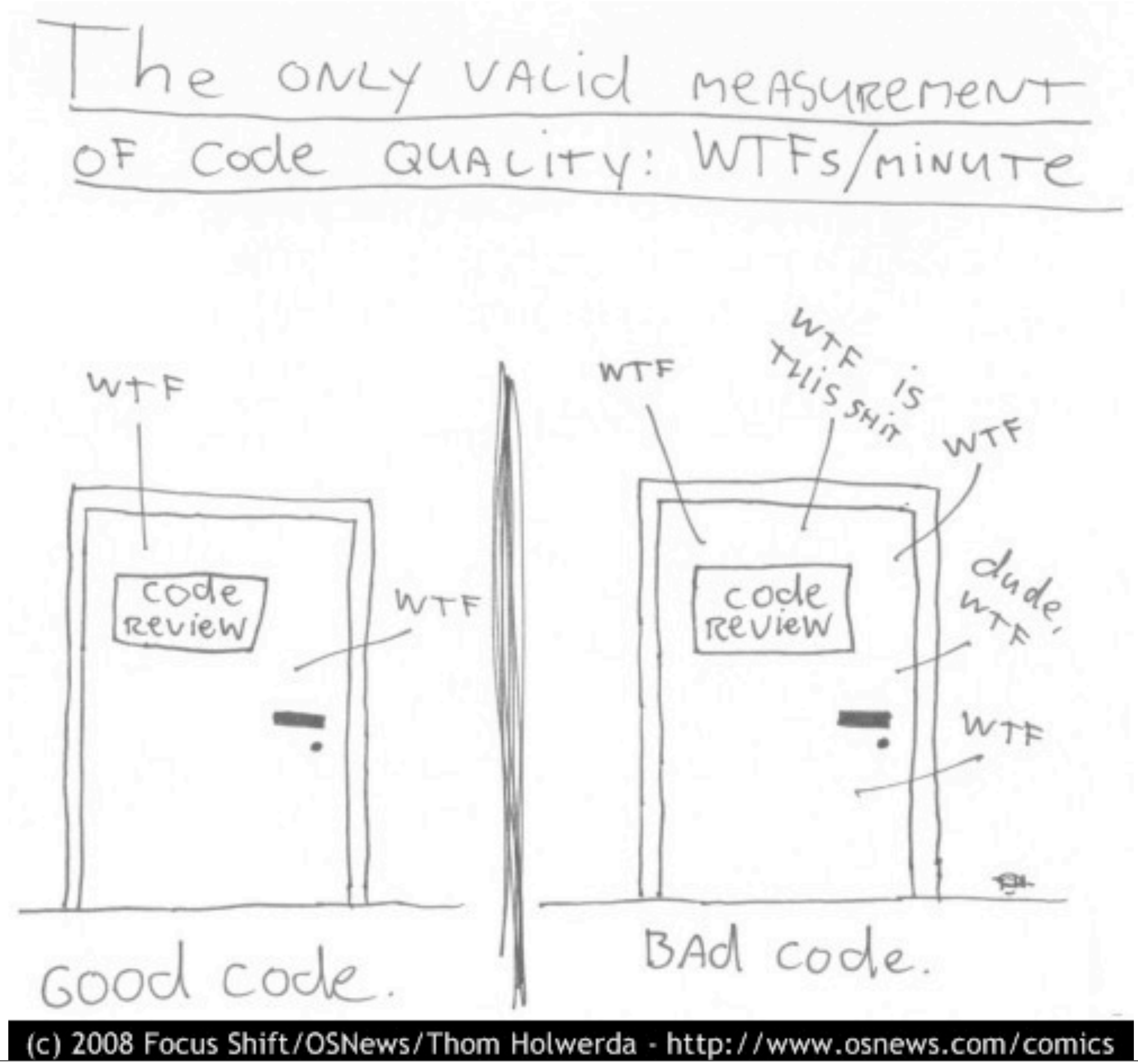
In Practice: What's in a review?



(c) 2008 Focus Shift/OSNews/Thom Holwerda - http://www.osnews.com/comics

- **At least one competent reviewer**
- **Early feedback with opportunity for followup**

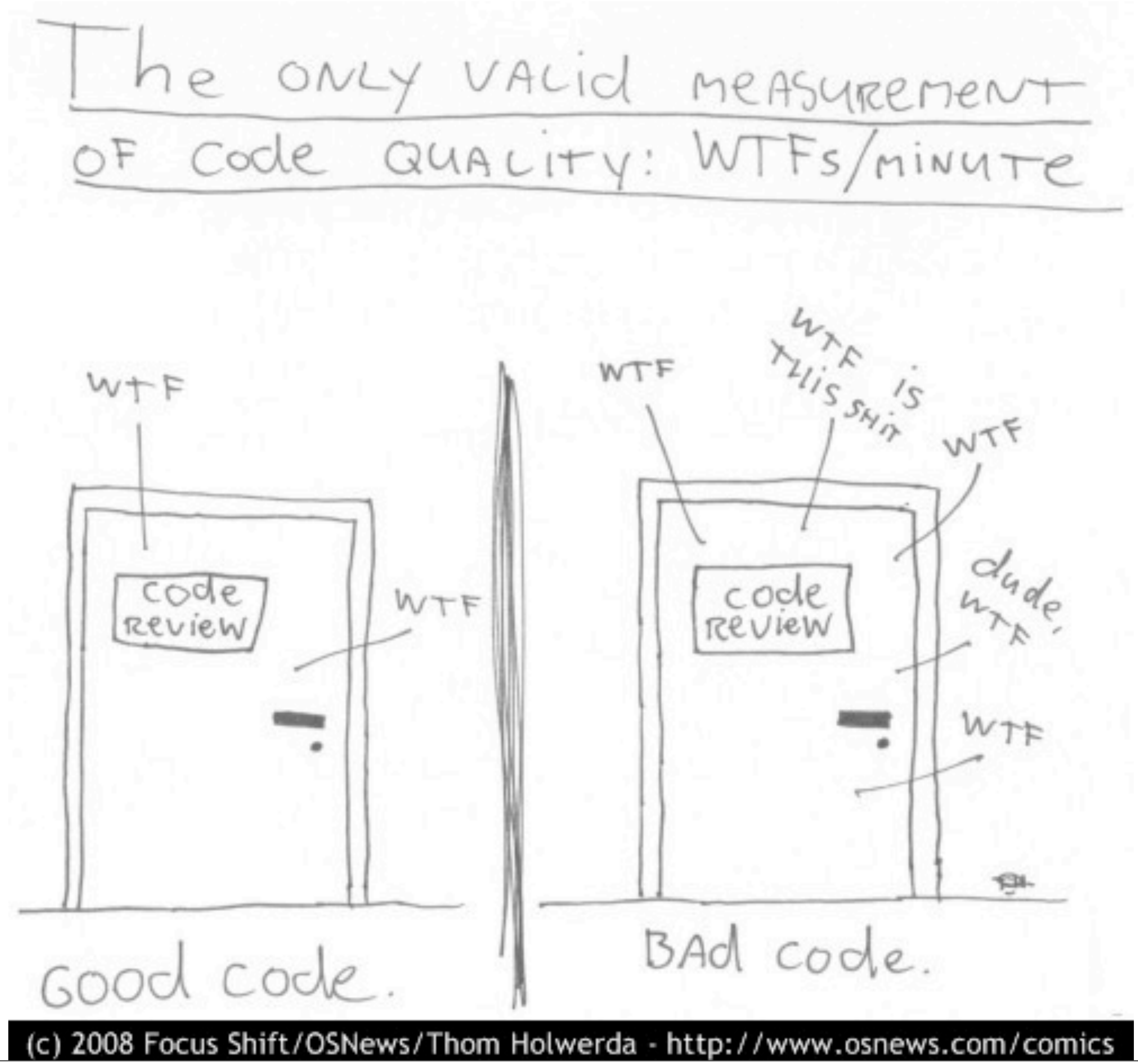
In Practice: What's in a review?



(c) 2008 Focus Shift/OSNews/Thom Holwerda - http://www.osnews.com/comics

- At least one competent reviewer
- Early feedback with opportunity for followup
- Review before "committing"

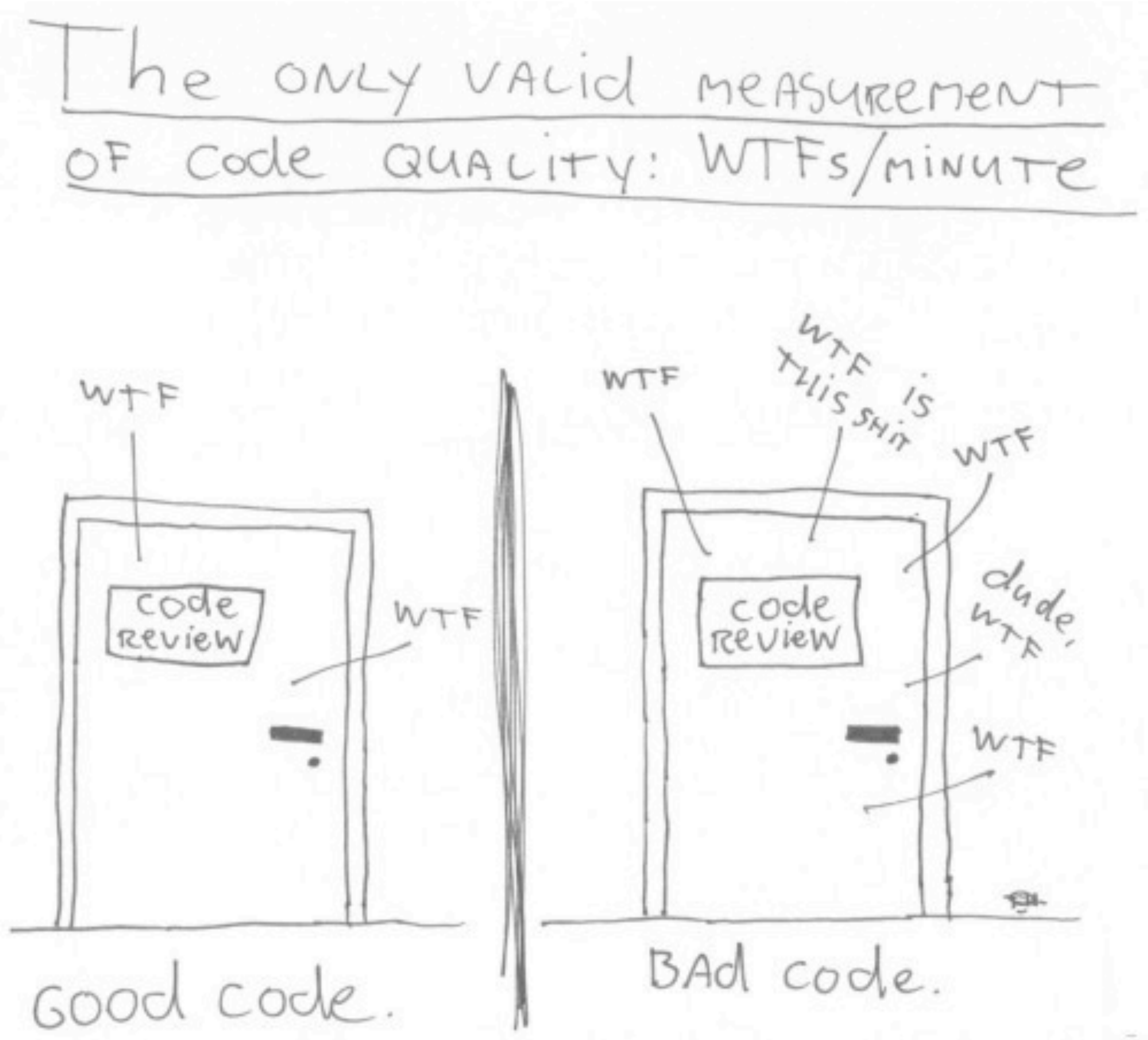
In Practice: What's in a review?



(c) 2008 Focus Shift/OSNews/Thom Holwerda - http://www.osnews.com/comics

- At least one competent reviewer
- Early feedback with opportunity for followup
- Review before "committing"
- Reviewer can block commit

In Practice: What's in a review?



(c) 2008 Focus Shift/OSNews/Thom Holwerda - http://www.osnews.com/comics

- **At least one competent reviewer**
- **Early feedback with opportunity for followup**
- **Review before "committing"**
- **Reviewer can block commit**
- **Author has final say on commit**

References

“Best Kept Secrets of Peer Code Review”, Jason Cohen et al.

“Peer Reviews in Software: A Practical Guide”, Karl E. Wiegers

"Facts & Fallacies of Software Engineering", Robert Glass

“Peopleware”, Tom DeMarco, Tim Lister

“The Goal”, Eli Goldratt

“Software Defect Reduction Top 10 List”, Barry Boehm, Victor R. Basili
http://www.cs.umd.edu/projects/SoftEng/ESEG/papers/82.78.pdf

http://www.ibm.com/developerworks/rational/library/11-proven-practices-for-peer-review/

http://svenpet.com/2014/01/07/better-code-reviews/

http://phinze.github.io/2013/12/08/pairing-vs-code-review.html

Thank you!

Austin Bingham

@austin_bingham

SixtyNORTH

@sixty_north